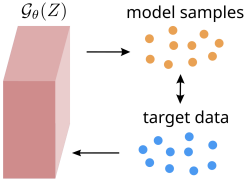
Generative Diffusion Methods

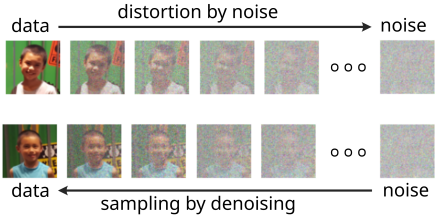
Paul J. Atzberger

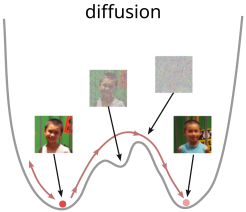
260J: Machine Learning
University of California Santa Barbara

Introduction

Generative Modeling

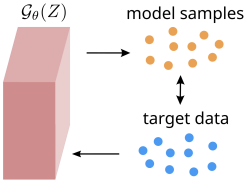


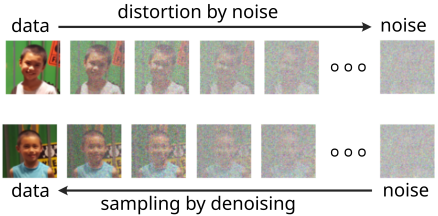


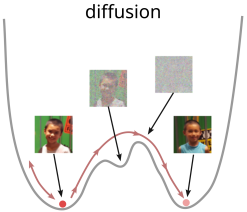


Introduction

Generative Modeling







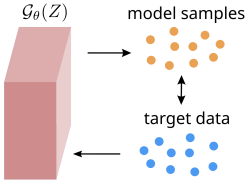
Motivations and History

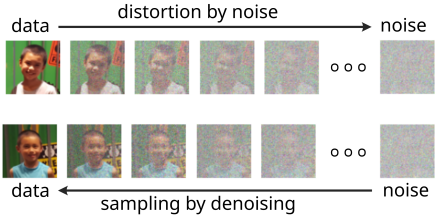
How can neural networks generate samples similar to a target data distribution μ_X?

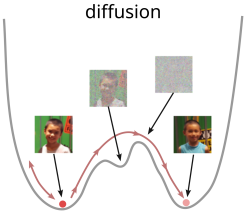
$\tilde{X} = \mathcal{G}_\theta(Z)$, to obtain $\tilde{X} \sim \mu_X$, with Z noise.

Introduction

Generative Modeling







Motivations and History

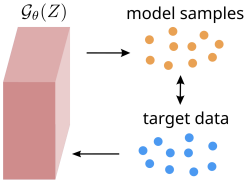
How can neural networks generate samples similar to a target data distribution μ_X?

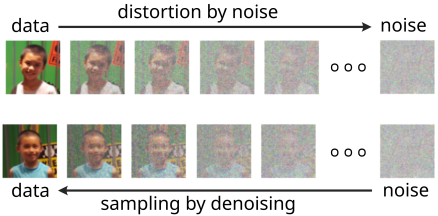
$\tilde{X} = \mathcal{G}_\theta(Z)$, to obtain $\tilde{X} \sim \mu_X$, with Z noise.

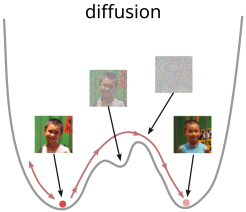
Tasks: AI Image Generation, Video Creation, Natural Language Summaries.

Introduction

Generative Modeling







Motivations and History

How can neural networks generate samples similar to a target data distribution μ_X?

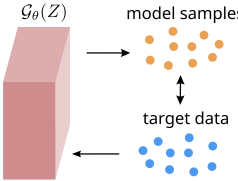
$\tilde{X} = \mathcal{G}_\theta(Z)$, to obtain $\tilde{X} \sim \mu_X$, with Z noise.

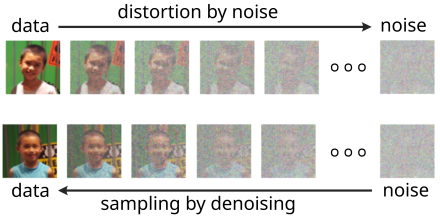
Tasks: AI Image Generation, Video Creation, Natural Language Summaries.

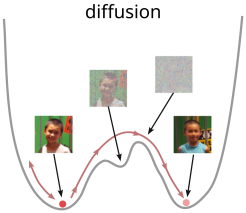
Directly using probability densities $p(\mathbf{x};\theta) = \frac{q(\mathbf{x};\theta)}{\mathcal{Z}(\theta)}$ involves local weight function q and normalization $\mathcal{Z}$ so $p(x)$ integrates to one.

Introduction

Generative Modeling







Motivations and History

How can neural networks generate samples similar to a target data distribution μ_X?

$\tilde{X} = \mathcal{G}_\theta(Z)$, to obtain $\tilde{X} \sim \mu_X$, with Z noise.

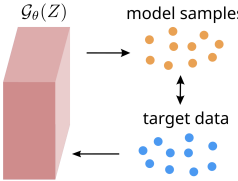
Tasks: AI Image Generation, Video Creation, Natural Language Summaries.

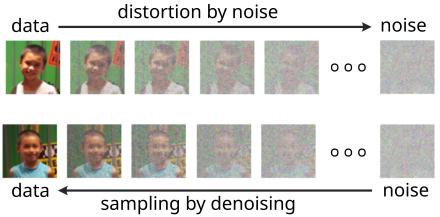
Directly using probability densities $p(\mathbf{x};\theta) = \frac{q(\mathbf{x};\theta)}{\mathcal{Z}(\theta)}$ involves local weight function q and normalization $\mathcal{Z}$ so $p(x)$ integrates to one.

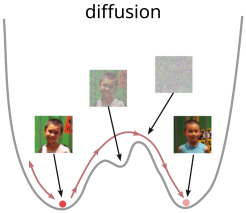
Maximum Likelihood (MLE) Methods in principle can fit observed data to obtain model distribution p_{θ^*}, $\theta^* = \arg\min_\theta D_{KL}(\mu_X | p_\theta)$.

Introduction

Generative Modeling







Motivations and History

How can neural networks generate samples similar to a target data distribution μ_X?

$\tilde{X} = \mathcal{G}_\theta(Z)$, to obtain $\tilde{X} \sim \mu_X$, with Z noise.

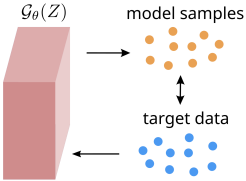
Tasks: AI Image Generation, Video Creation, Natural Language Summaries.

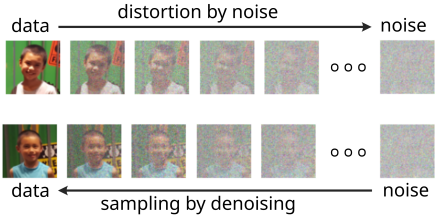
Directly using probability densities $p(\mathbf{x};\theta) = \frac{q(\mathbf{x};\theta)}{\mathcal{Z}(\theta)}$ involves local weight function q and normalization $\mathcal{Z}$ so $p(x)$ integrates to one.

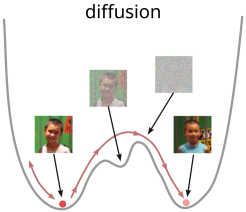
Maximum Likelihood (MLE) Methods in principle can fit observed data to obtain model distribution p_{θ^*}, $\theta^* = \arg\min_\theta D_{KL}(\mu_X | p_\theta)$.

Introduction

Generative Modeling







Motivations and History

How can neural networks generate samples similar to a target data distribution μ_X?

$\tilde{X} = \mathcal{G}_\theta(Z)$, to obtain $\tilde{X} \sim \mu_X$, with Z noise.

Tasks: AI Image Generation, Video Creation, Natural Language Summaries.

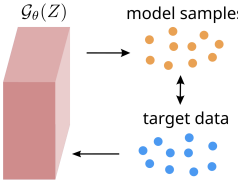
Directly using probability densities $p(\mathbf{x};\theta) = \frac{q(\mathbf{x};\theta)}{\mathcal{Z}(\theta)}$ involves local weight function q and normalization $\mathcal{Z}$ so $p(x)$ integrates to one.

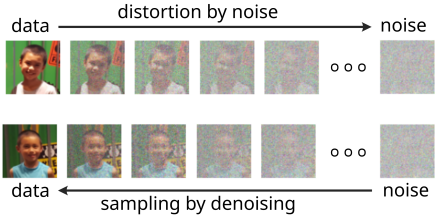
Maximum Likelihood (MLE) Methods in principle can fit observed data to obtain model distribution p_{θ^*}, $\theta^* = \arg\min_\theta D_{KL}(\mu_X | p_\theta)$.

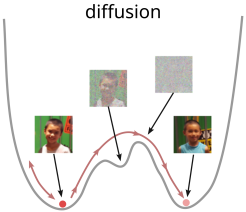
However, direct MLE requires densities with normalization $\mathcal{Z}$ which poses difficulties in practice.

Introduction

Generative Modeling







Motivations and History

How can neural networks generate samples similar to a target data distribution μ_X?

$\tilde{X} = \mathcal{G}_\theta(Z)$, to obtain $\tilde{X} \sim \mu_X$, with Z noise.

Tasks: AI Image Generation, Video Creation, Natural Language Summaries.

Directly using probability densities $p(\mathbf{x};\theta) = \frac{q(\mathbf{x};\theta)}{\mathcal{Z}(\theta)}$ involves local weight function q and normalization $\mathcal{Z}$ so $p(x)$ integrates to one.

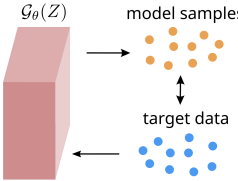
Maximum Likelihood (MLE) Methods in principle can fit observed data to obtain model distribution p_{θ^*}, $\theta^* = \arg\min_\theta D_{KL}(\mu_X | p_\theta)$.

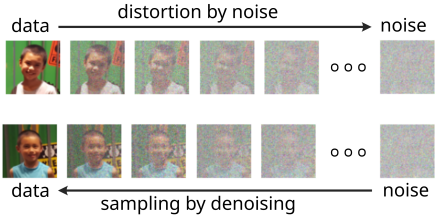
However, direct MLE requires densities with normalization $\mathcal{Z}$ which poses difficulties in practice.

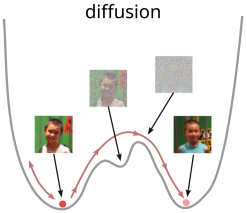
Alternative: Implicit Generative Approaches: $\mathcal{G}_\theta$ generates samples $\tilde{X}$ from noise Z.

Introduction

Generative Modeling







Motivations and History

How can neural networks generate samples similar to a target data distribution μ_X?

$\tilde{X} = \mathcal{G}_\theta(Z)$, to obtain $\tilde{X} \sim \mu_X$, with Z noise.

Tasks: AI Image Generation, Video Creation, Natural Language Summaries.

Directly using probability densities $p(\mathbf{x};\theta) = \frac{q(\mathbf{x};\theta)}{\mathcal{Z}(\theta)}$ involves local weight function q and normalization $\mathcal{Z}$ so $p(x)$ integrates to one.

Maximum Likelihood (MLE) Methods in principle can fit observed data to obtain model distribution p_{θ^*}, $\theta^* = \arg\min_\theta D_{KL}(\mu_X | p_\theta)$.

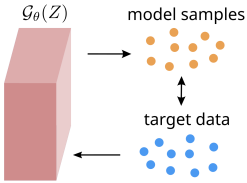
However, direct MLE requires densities with normalization $\mathcal{Z}$ which poses difficulties in practice.

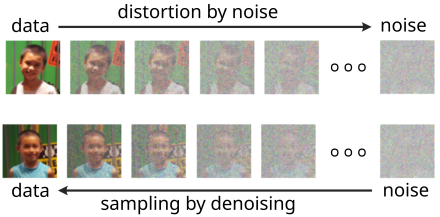
Alternative: Implicit Generative Approaches: $\mathcal{G}_\theta$ generates samples $\tilde{X}$ from noise Z.

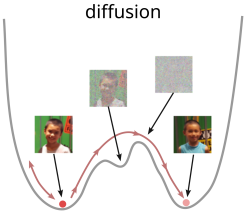
Generative Adversarial Networks (GANs) (Goodfellow 2014) provide ways to train implicit generators (however, can be unstable and exhibit mode collapse).

Introduction

Generative Modeling



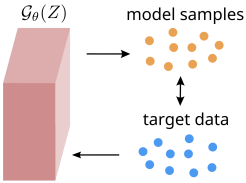


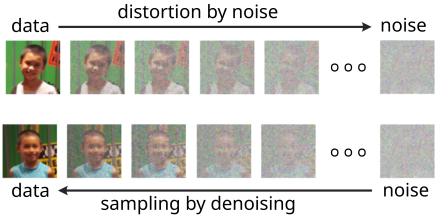


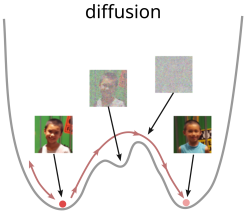
Motivations and History

Introduction

Generative Modeling





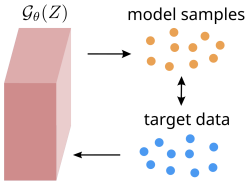


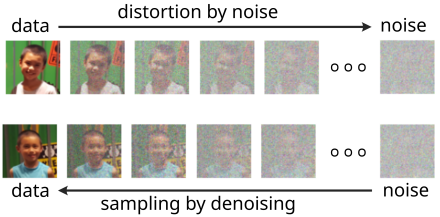
Motivations and History

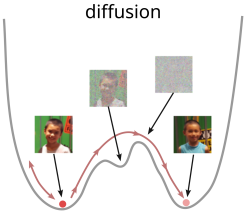
Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Introduction

Generative Modeling







Motivations and History

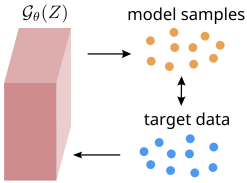
Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

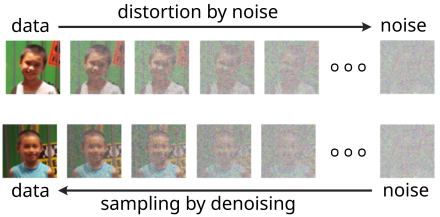
Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

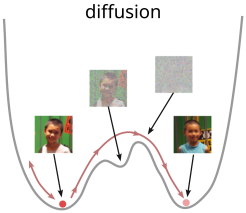
$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

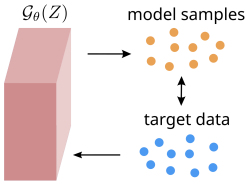
Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

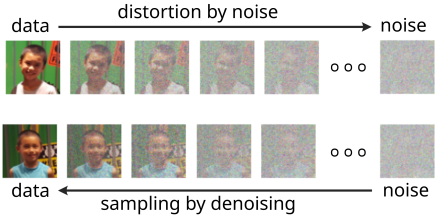
$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

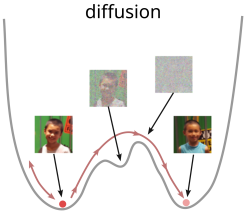
This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

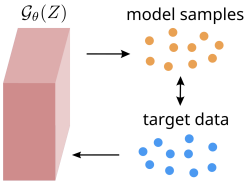
Normalization is not required since $p = q/\mathcal{Z}$,

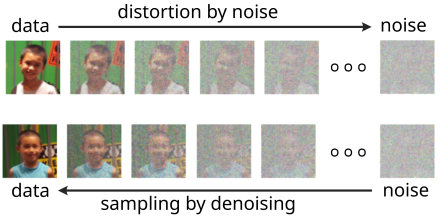
$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

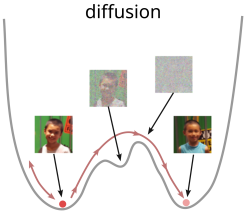
$-\nabla_x u$ does not depend on the additive constant.

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

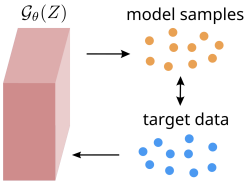
$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

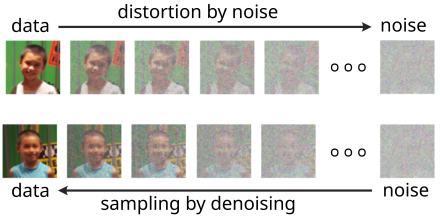
$-\nabla_x u$ does not depend on the additive constant.

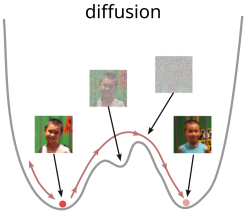
How can this be done without density p_X?

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

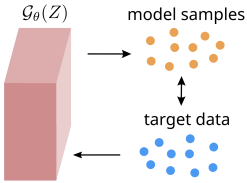
$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

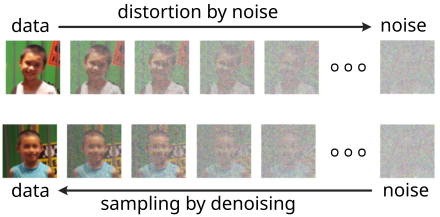
$-\nabla_x u$ does not depend on the additive constant.

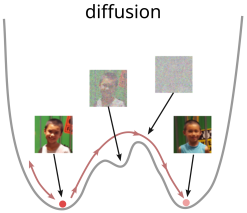
How can this be done without density p_X?

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

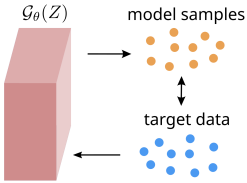
$-\nabla_x u$ does not depend on the additive constant.

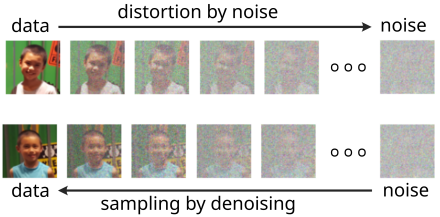
How can this be done without density p_X?

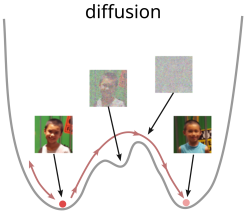
Hyvärinen 2005 derived useful identity for estimating gradients of the log-probability of empirical distributions (only needs sampled data).

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

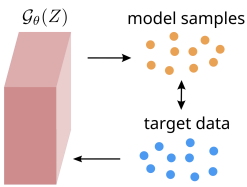
$-\nabla_x u$ does not depend on the additive constant.

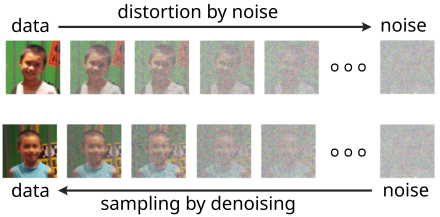
How can this be done without density p_X?

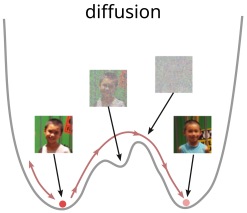
Hyvärinen 2005 derived useful identity for estimating gradients of the log-probability of empirical distributions (only needs sampled data).

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

$-\nabla_x u$ does not depend on the additive constant.

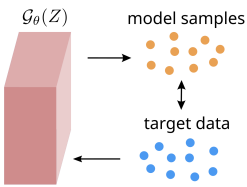
How can this be done without density p_X?

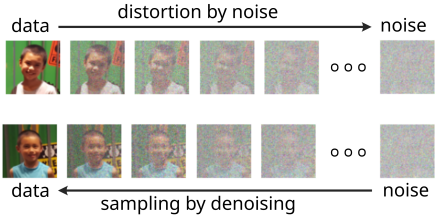
Hyvärinen 2005 derived useful identity for estimating gradients of the log-probability of empirical distributions (only needs sampled data).

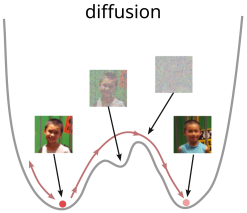
Denoising Diffusion Probabilistic Model (DDPM) (Sohl-Dickstein et al. 2015) (Ho 2020) motivated by diffusive sampling in non-equil statistical mechanics.

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

$-\nabla_x u$ does not depend on the additive constant.

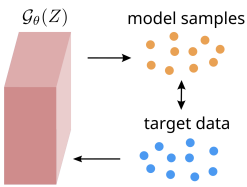
How can this be done without density p_X?

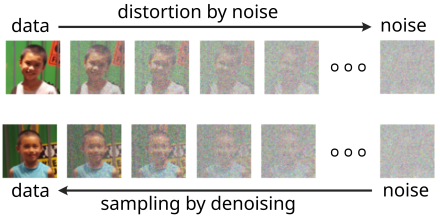
Hyvärinen 2005 derived useful identity for estimating gradients of the log-probability of empirical distributions (only needs sampled data).

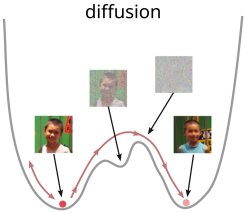
Denoising Diffusion Probabilistic Model (DDPM) (Sohl-Dickstein et al. 2015) (Ho 2020) motivated by diffusive sampling in non-equil statistical mechanics.

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

$-\nabla_x u$ does not depend on the additive constant.

How can this be done without density p_X?

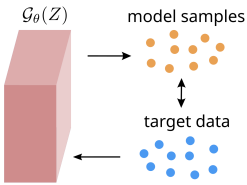
Hyvärinen 2005 derived useful identity for estimating gradients of the log-probability of empirical distributions (only needs sampled data).

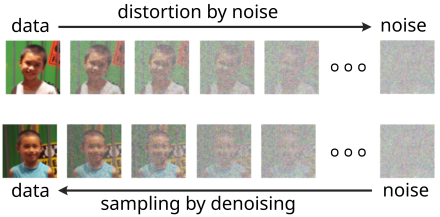
Denoising Diffusion Probabilistic Model (DDPM) (Sohl-Dickstein et al. 2015) (Ho 2020) motivated by diffusive sampling in non-equil statistical mechanics.

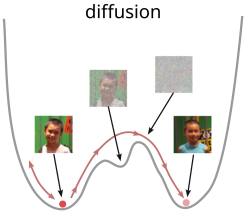
DDPM: Two stages:

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

$-\nabla_x u$ does not depend on the additive constant.

How can this be done without density p_X?

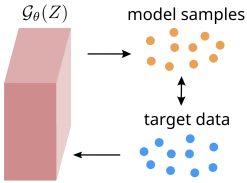
Hyvärinen 2005 derived useful identity for estimating gradients of the log-probability of empirical distributions (only needs sampled data).

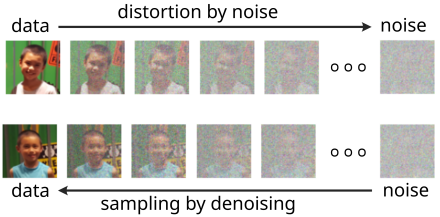
Denoising Diffusion Probabilistic Model (DDPM) (Sohl-Dickstein et al. 2015) (Ho 2020) motivated by diffusive sampling in non-equil statistical mechanics.

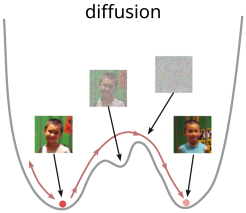
DDPM: Two stages:
(i) noise is added to target distribution μ_X to obtain a canonical distribution μ_Z,

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

$-\nabla_x u$ does not depend on the additive constant.

How can this be done without density p_X?

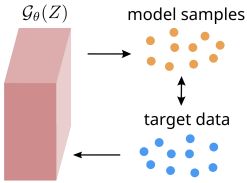
Hyvärinen 2005 derived useful identity for estimating gradients of the log-probability of empirical distributions (only needs sampled data).

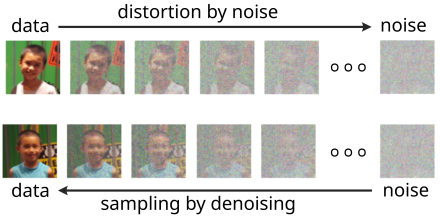
Denoising Diffusion Probabilistic Model (DDPM) (Sohl-Dickstein et al. 2015) (Ho 2020) motivated by diffusive sampling in non-equil statistical mechanics.

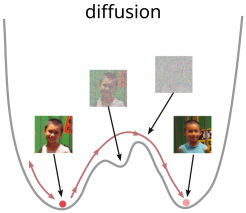
DDPM: Two stages:
(i) noise is added to target distribution μ_X to obtain a canonical distribution μ_Z,
(ii) de-noising is learned to reverse this process to recover the original distribution μ_X.

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

$-\nabla_x u$ does not depend on the additive constant.

How can this be done without density p_X?

Hyvärinen 2005 derived useful identity for estimating gradients of the log-probability of empirical distributions (only needs sampled data).

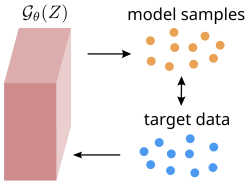
Denoising Diffusion Probabilistic Model (DDPM) (Sohl-Dickstein et al. 2015) (Ho 2020) motivated by diffusive sampling in non-equil statistical mechanics.

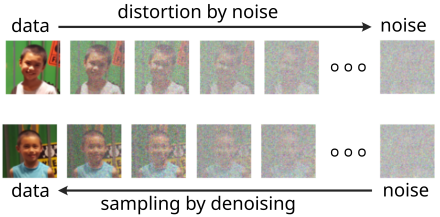
DDPM: Two stages:
(i) noise is added to target distribution μ_X to obtain a canonical distribution μ_Z,
(ii) de-noising is learned to reverse this process to recover the original distribution μ_X.

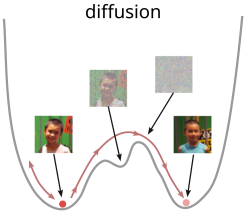
Demonstrated how Gaussian noise can be used to sample complicated distributions.

Introduction

Generative Modeling







Motivations and History

Diffusion Methods learn a stochastic process X_t whose invariant probability is a target distribution $\mu_X \sim p_X$, i.e. $p_X = p_{data}$.

Strategy: For $p_X(x)$ learn a potential energy $u(x) = -\log(p_X(x)) + C$ for Langevin dynamics

$dX_t = -\nabla_x u(X_t)dt + \sqrt{2}dW_t$.

This has the Gibbs-Boltzmann invariant distribution $p(x) \propto \exp(-u(x)) \Rightarrow p(x) = p_X(x)$.

Normalization is not required since $p = q/\mathcal{Z}$,

$u(x) = -\log(p(x)) = -\log(q(x)) + \log(\mathcal{Z})$ and

$-\nabla_x u$ does not depend on the additive constant.

How can this be done without density p_X?

Hyvärinen 2005 derived useful identity for estimating gradients of the log-probability of empirical distributions (only needs sampled data).

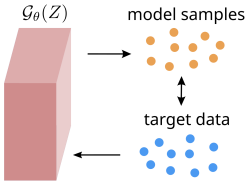
Denoising Diffusion Probabilistic Model (DDPM) (Sohl-Dickstein et al. 2015) (Ho 2020) motivated by diffusive sampling in non-equil statistical mechanics.

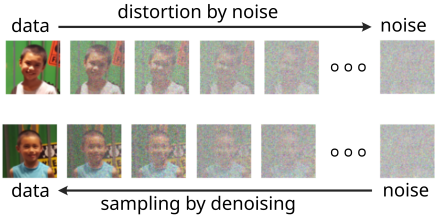
DDPM: Two stages:
(i) noise is added to target distribution μ_X to obtain a canonical distribution μ_Z,
(ii) de-noising is learned to reverse this process to recover the original distribution μ_X.

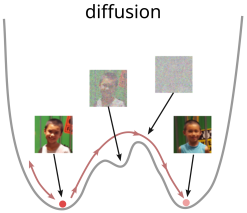
Demonstrated how Gaussian noise can be used to sample complicated distributions.

Introduction

Generative Modeling



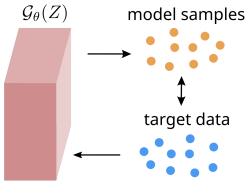


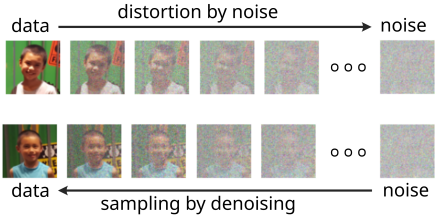


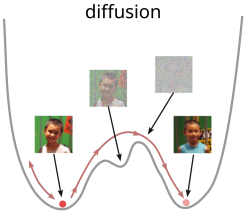
Motivations and History

Introduction

Generative Modeling





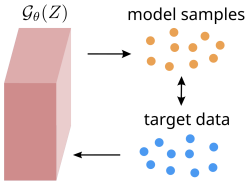


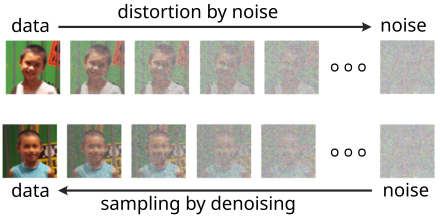
Motivations and History

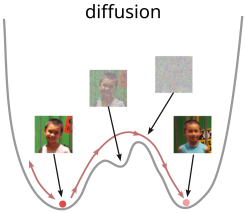
Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

Introduction

Generative Modeling







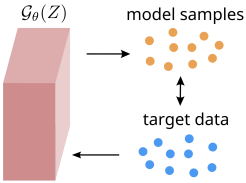
Motivations and History

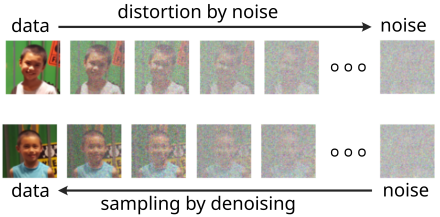
Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

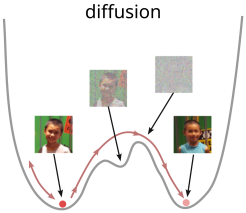
Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

Introduction

Generative Modeling







Motivations and History

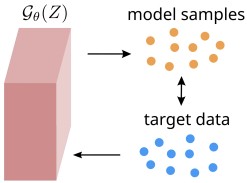
Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

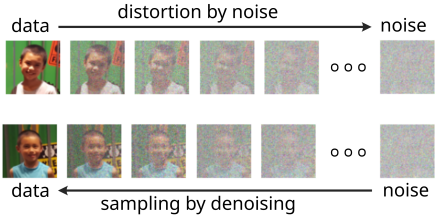
Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

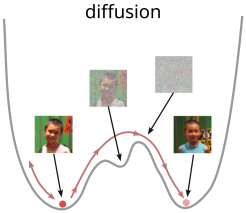
Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

Introduction

Generative Modeling







Motivations and History

Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

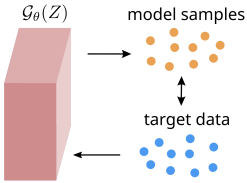
Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

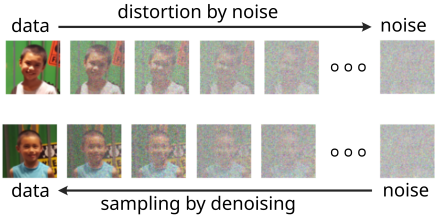
Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

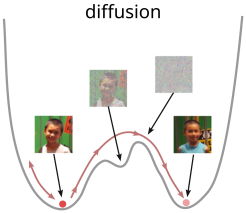
Sampling is performed by diffusion from solving numerically the SDEs.

Introduction

Generative Modeling







Motivations and History

Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

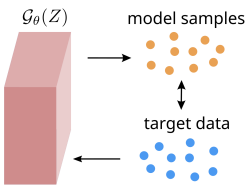
Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

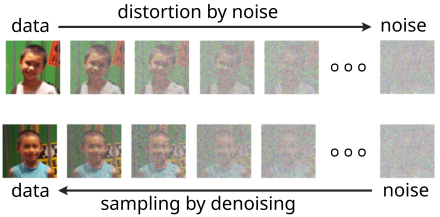
Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

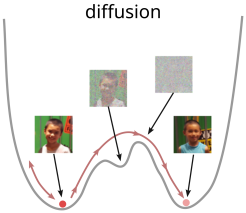
Sampling is performed by diffusion from solving numerically the SDEs.

Introduction

Generative Modeling







Motivations and History

Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

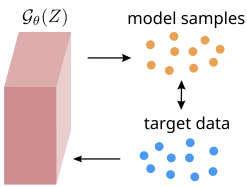
Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

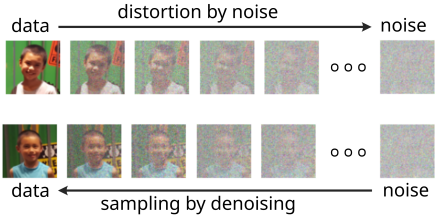
Sampling is performed by diffusion from solving numerically the SDEs.

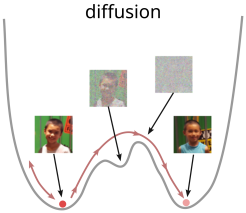
High-dimensional probability distributions can be effectively sampled with these methods.

Introduction

Generative Modeling







Motivations and History

Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

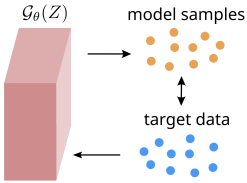
Sampling is performed by diffusion from solving numerically the SDEs.

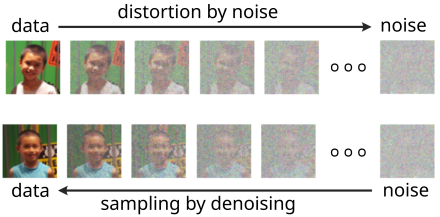
High-dimensional probability distributions can be effectively sampled with these methods.

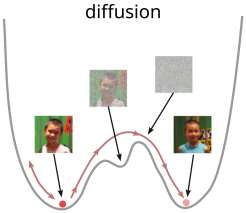
Allows for sampling images, audio, and even video sequences. Many potential applications.

Introduction

Generative Modeling







Motivations and History

Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

Sampling is performed by diffusion from solving numerically the SDEs.

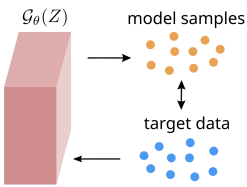
High-dimensional probability distributions can be effectively sampled with these methods.

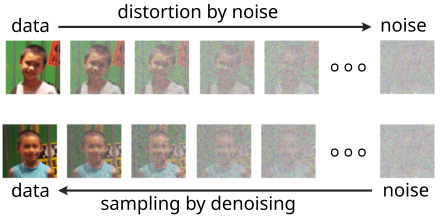
Allows for sampling images, audio, and even video sequences. Many potential applications.

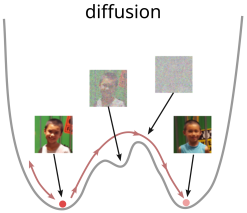
Active area of research and development.

Introduction

Generative Modeling







Motivations and History

Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

Sampling is performed by diffusion from solving numerically the SDEs.

High-dimensional probability distributions can be effectively sampled with these methods.

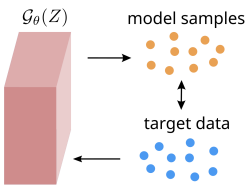
Allows for sampling images, audio, and even video sequences. Many potential applications.

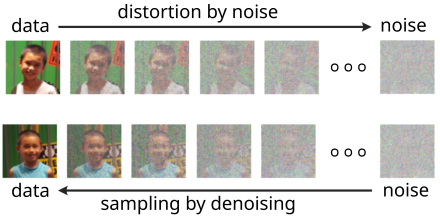
Active area of research and development.

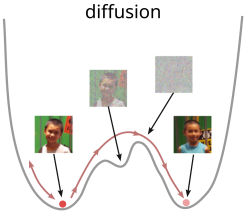
Scalability has resulted in current era of generative AI services.

Introduction

Generative Modeling







Motivations and History

Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

Sampling is performed by diffusion from solving numerically the SDEs.

High-dimensional probability distributions can be effectively sampled with these methods.

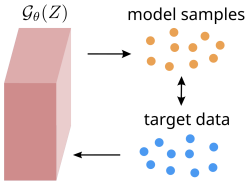
Allows for sampling images, audio, and even video sequences. Many potential applications.

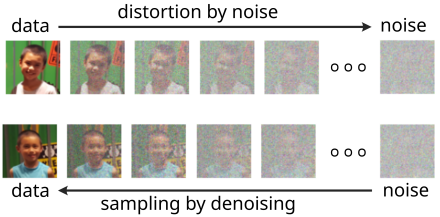
Active area of research and development.

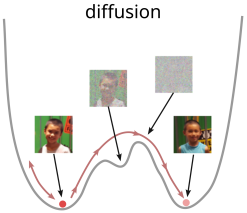
Scalability has resulted in current era of generative AI services.

Introduction

Generative Modeling







Motivations and History

Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

Sampling is performed by diffusion from solving numerically the SDEs.

High-dimensional probability distributions can be effectively sampled with these methods.

Allows for sampling images, audio, and even video sequences. Many potential applications.

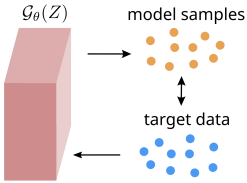
Active area of research and development.

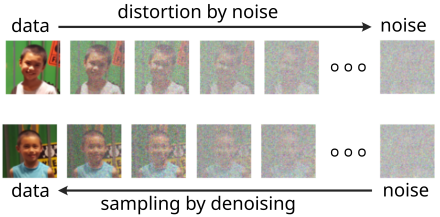
Scalability has resulted in current era of generative AI services.

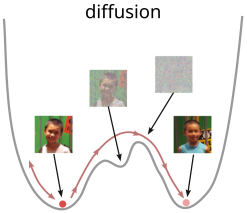
Crucial components in Midjourney, DALL-E, Google Gemini, and other services.

Introduction

Generative Modeling







Motivations and History

Noise Conditional Score Networks (NCSN) (Song & Ermon 2019) uses noise at different levels to obtain annealed Langevin dynamics to sample complex high-dimensional distributions.

Score Matching SDEs (Song, Sohl-Dickstein, et al. 2020) unifies approaches using the score function $s_\theta(x) = \nabla \log(p_\theta(x))$ to show how a general class of SDEs can be reversed.

Challenge reduces to learning the score function $s_\theta(x)$ using deep neural networks.

Sampling is performed by diffusion from solving numerically the SDEs.

High-dimensional probability distributions can be effectively sampled with these methods.

Allows for sampling images, audio, and even video sequences. Many potential applications.

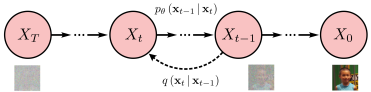
Active area of research and development.

Scalability has resulted in current era of generative AI services.

Crucial components in Midjourney, DALL-E, Google Gemini, and other services.

Denoising Diffusion Probabilistic Model (DDPM)

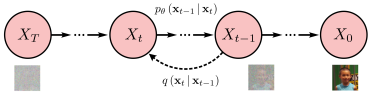
Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



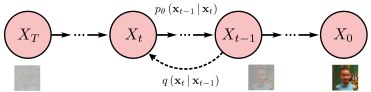
DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right).$

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

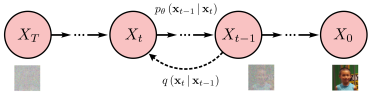
Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

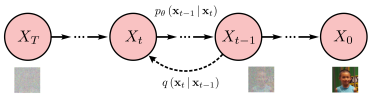
$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right).$

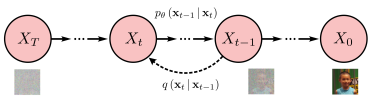
Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow\ q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \,|\, \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right).$

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

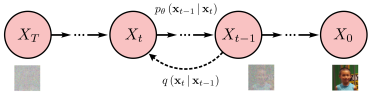
Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow\ q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

Backward diffusion process aims to reverse this by learning a Markov chain with transitions $p_\theta(\mathbf{x}_{t-1} \,|\, \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$.

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t\,|\,\mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

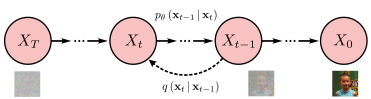
For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow\ q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

Backward diffusion process aims to reverse this by learning a Markov chain with transitions $p_\theta(\mathbf{x}_{t-1}\,|\,\mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$.

Evidence Lower Bound (ELBO) bounds below the log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$, $\mathrm{ELBO}(\theta) = \mathbb{E}_q\left[\log\left(p(\mathbf{x}_T)\right) + \sum_{t=1}^{T} \log\left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right)\right]$.

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow\ q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

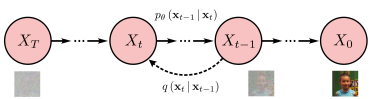
Backward diffusion process aims to reverse this by learning a Markov chain with transitions $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$.

Evidence Lower Bound (ELBO) bounds below the log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$, $\mathrm{ELBO}(\theta) = \mathbb{E}_q\left[\log\left(p(\mathbf{x}_T)\right) + \sum_{t=1}^{T} \log\left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right)\right]$.

Loss: $\ell(\theta) = -\mathrm{ELBO}(\theta)$, maximizes log-likelihood of Gaussians p_θ using ELBO. Allows for closed-form expressions and tractable computations.

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow\ q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

Backward diffusion process aims to reverse this by learning a Markov chain with transitions $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$.

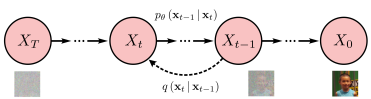
Evidence Lower Bound (ELBO) bounds below the log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$, $\mathrm{ELBO}(\theta) = \mathbb{E}_q\left[\log\left(p(\mathbf{x}_T)\right) + \sum_{t=1}^{T} \log\left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right)\right]$.

Loss: $\ell(\theta) = -\mathrm{ELBO}(\theta)$, maximizes log-likelihood of Gaussians p_θ using ELBO. Allows for closed-form expressions and tractable computations.

Case $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathcal{I}$**:** Loss can be shown to become

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow\ q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

Backward diffusion process aims to reverse this by learning a Markov chain with transitions $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$.

Evidence Lower Bound (ELBO) bounds below the log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$, $\mathrm{ELBO}(\theta) = \mathbb{E}_q\left[\log\left(p(\mathbf{x}_T)\right) + \sum_{t=1}^{T} \log\left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right)\right]$.

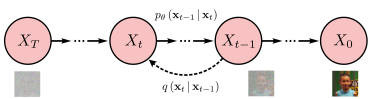
Loss: $\ell(\theta) = -\mathrm{ELBO}(\theta)$, maximizes log-likelihood of Gaussians p_θ using ELBO. Allows for closed-form expressions and tractable computations.

Case $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathcal{I}$**:** Loss can be shown to become $\ell(\theta) =$

$$\mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right] + C.$$

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow\ q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

Backward diffusion process aims to reverse this by learning a Markov chain with transitions $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$.

Evidence Lower Bound (ELBO) bounds below the log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$, $\mathrm{ELBO}(\theta) = \mathbb{E}_q\left[\log\left(p(\mathbf{x}_T)\right) + \sum_{t=1}^{T} \log\left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right)\right]$.

Loss: $\ell(\theta) = -\mathrm{ELBO}(\theta)$, maximizes log-likelihood of Gaussians p_θ using ELBO. Allows for closed-form expressions and tractable computations.

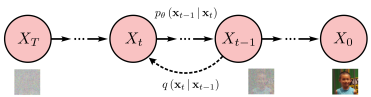
Case $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathcal{I}$**:** Loss can be shown to become $\ell(\theta) =$

$$\mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right] + C.$$

$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow \; q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

Backward diffusion process aims to reverse this by learning a Markov chain with transitions $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$.

Evidence Lower Bound (ELBO) bounds below the log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$,
$\mathrm{ELBO}(\theta) = \mathbb{E}_q\left[\log\left(p(\mathbf{x}_T)\right) + \sum_{t=1}^{T} \log\left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right)\right]$.

Loss: $\ell(\theta) = -\mathrm{ELBO}(\theta)$, maximizes log-likelihood of Gaussians p_θ using ELBO. Allows for closed-form expressions and tractable computations.

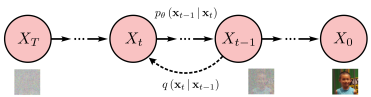
Case $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathcal{I}$**:** Loss can be shown to become $\ell(\theta) =$
$\mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right] + C$.

$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

Simplifying approximation (works well in practice)
$\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow\ q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

Backward diffusion process aims to reverse this by learning a Markov chain with transitions $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$.

Evidence Lower Bound (ELBO) bounds below the log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$,
$\mathrm{ELBO}(\theta) = \mathbb{E}_q\left[\log\left(p(\mathbf{x}_T)\right) + \sum_{t=1}^{T} \log\left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right)\right]$.

Loss: $\ell(\theta) = -\mathrm{ELBO}(\theta)$, maximizes log-likelihood of Gaussians p_θ using ELBO. Allows for closed-form expressions and tractable computations.

Case $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathcal{I}$**:** Loss can be shown to become $\ell(\theta) =$
$\mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right] + C$.

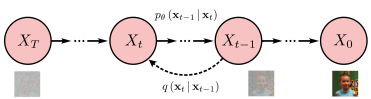
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

Simplifying approximation (works well in practice)
$\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.

(i) Forward-process provides training samples, (ii) Learned backward-process provides sampler.

Denoising Diffusion Probabilistic Model (DDPM)

Sampling by Denoising



DDPM samples by a denoising diffusion process (Sohl-Dickstein 2015) and (Ho 2020).

Forward diffusion process adds noise using Markov chain with transitions

$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\left(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t \mathcal{I}\right)$.

Maps target μ_X to a canonical distribution $\mu_Z \sim \mathcal{N}(\mathbf{z}; 0, \mathcal{I})$ (provides training samples).

Noise increments have scheduled variances given by $\beta_1, \beta_2, \ldots, \beta_T$. Factors $\bar{\mathbf{x}}_t = \sqrt{1-\beta_t}\bar{\mathbf{x}}_{t-1}$ drive mean $\bar{\mathbf{x}}_t$ toward 0 to obtain in long-time limit distribution μ_Z (conditions on β_t).

For step t let $\bar{\alpha}_t = \prod_{s=1}^{t} \alpha_s$ where $\alpha_t = 1 - \beta_t$, then $q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0, (1-\bar{\alpha}_t)\mathcal{I})$, **Yields,** $\bar{\alpha}_t \to 0$ as $t \to \infty \Rightarrow\ q_t \to \mu_Z = \mathcal{N}(0, \mathcal{I})$.

Backward diffusion process aims to reverse this by learning a Markov chain with transitions $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}\left(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t)\right)$.

Evidence Lower Bound (ELBO) bounds below the log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$, $\mathrm{ELBO}(\theta) = \mathbb{E}_q\left[\log\left(p(\mathbf{x}_T)\right) + \sum_{t=1}^{T} \log\left(\frac{p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right)\right]$.

Loss: $\ell(\theta) = -\mathrm{ELBO}(\theta)$, maximizes log-likelihood of Gaussians p_θ using ELBO. Allows for closed-form expressions and tractable computations.

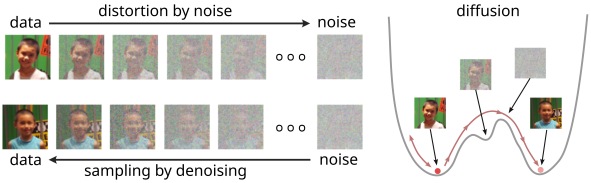
Case $\boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t) = \sigma_t^2 \mathcal{I}$**:** Loss can be shown to become $\ell(\theta) =$
$\mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\frac{\beta_t^2}{2\sigma_t^2 \alpha_t (1-\bar{\alpha}_t)} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right] + C$.

$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

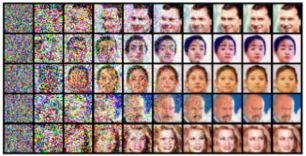
Simplifying approximation (works well in practice) $\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.

(i) Forward-process provides training samples, (ii) Learned backward-process provides sampler.

Score-Matching Diffusion Methods & SDEs



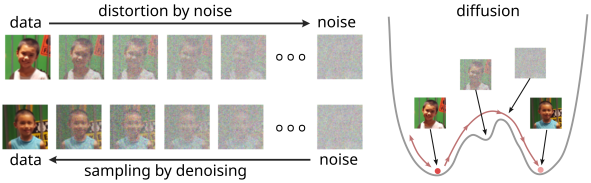
Annealed Sampling



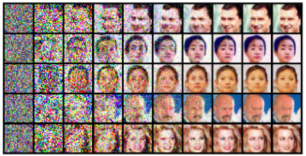
Song 2020

Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Score-Matching Diffusion Methods & SDEs



Annealed Sampling

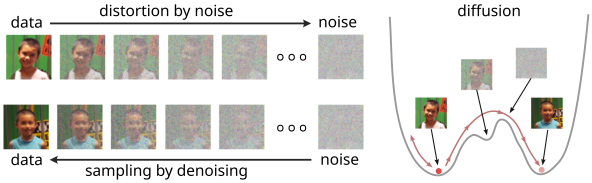


Song 2020

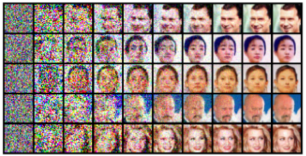
Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Score-Matching Diffusion Methods & SDEs



Annealed Sampling



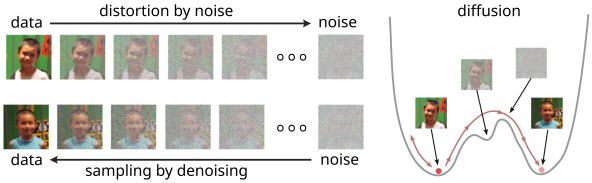
Song 2020

Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

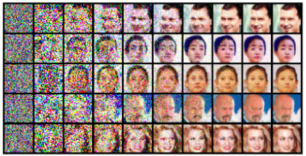
Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Score-Matching Diffusion Methods & SDEs



Annealed Sampling



Song 2020

Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

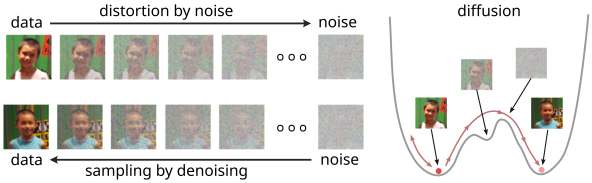
Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

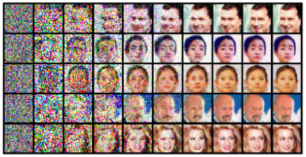
Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Score-Matching Diffusion Methods & SDEs



Annealed Sampling



Song 2020

Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

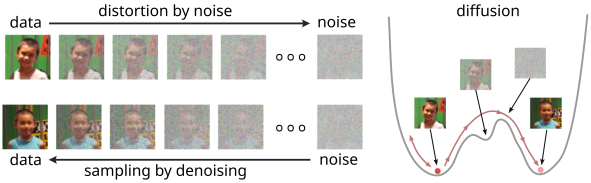
Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

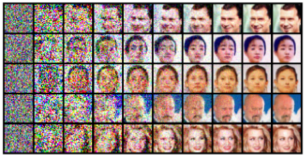
$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies:

Score-Matching Diffusion Methods & SDEs



Annealed Sampling



Song 2020

Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

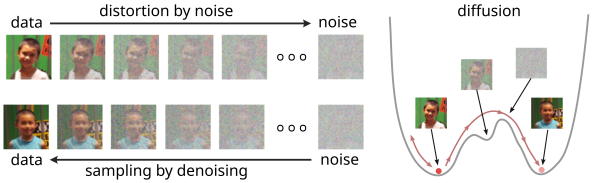
Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

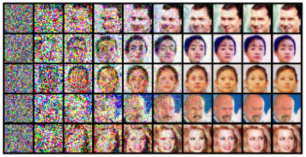
$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$,

Score-Matching Diffusion Methods & SDEs



Annealed Sampling



Song 2020

Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

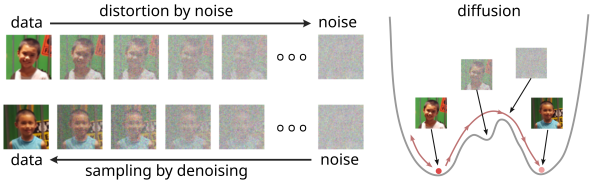
Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

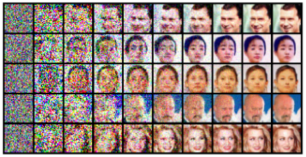
$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$, (ii) denoising score matching using $\tilde{s}(\mathbf{x}) = \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\tilde{\mathbf{x}}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Score-Matching Diffusion Methods & SDEs



Annealed Sampling



Song 2020

Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

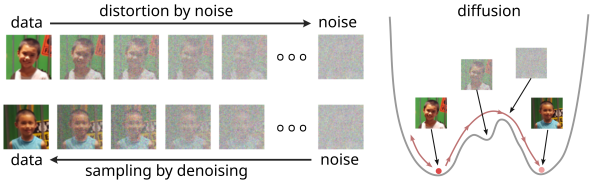
Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$, (ii) denoising score matching using $\tilde{s}(\mathbf{x}) = \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\tilde{\mathbf{x}}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Noise mitigates when density on low-dim manifold.

Score-Matching Diffusion Methods & SDEs



Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

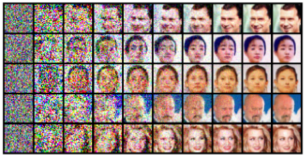
Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$, (ii) denoising score matching using $\tilde{s}(\mathbf{x}) = \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\tilde{\mathbf{x}}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Noise mitigates when density on low-dim manifold.

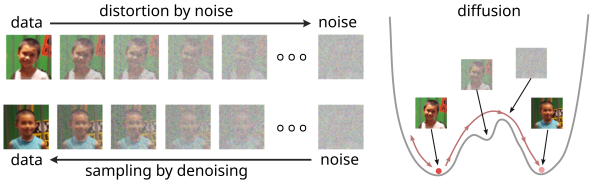
Annealed Sampling



Song 2020

Multi-level variant of (ii): Use several levels of Gaussian noise $\{\sigma_i\}_{i=1}^L$ and simultaneously estimates scores at all noise levels $s_\theta(\mathbf{x}, \sigma)$, (Song 2019).

Score-Matching Diffusion Methods & SDEs



Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

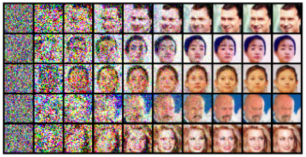
Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$, (ii) denoising score matching using $\tilde{s}(\mathbf{x}) = \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\tilde{\mathbf{x}}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Noise mitigates when density on low-dim manifold.

Annealed Sampling

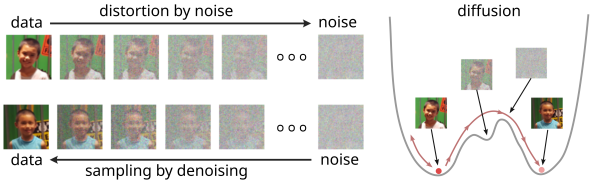


Song 2020

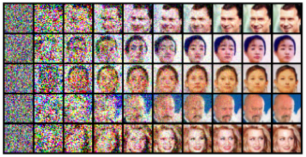
Multi-level variant of (ii): Use several levels of Gaussian noise $\{\sigma_i\}_{i=1}^L$ and simultaneously estimates scores at all noise levels $s_\theta(\mathbf{x}, \sigma)$, (Song 2019).

Noise Conditional Score Networks (NCSN):

Score-Matching Diffusion Methods & SDEs



Annealed Sampling



Song 2020

Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

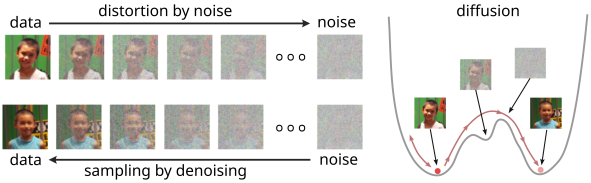
Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$, (ii) denoising score matching using $\tilde{s}(\mathbf{x}) = \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\tilde{\mathbf{x}}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Noise mitigates when density on low-dim manifold.

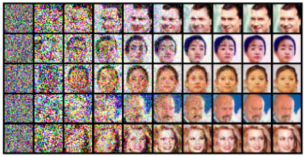
Multi-level variant of (ii): Use several levels of Gaussian noise $\{\sigma_i\}_{i=1}^L$ and simultaneously estimates scores at all noise levels $s_\theta(\mathbf{x}, \sigma)$, (Song 2019).

Noise Conditional Score Networks (NCSN): $s_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2 \mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Score-Matching Diffusion Methods & SDEs



Annealed Sampling



Song 2020

Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$, (ii) denoising score matching using $\tilde{s}(\mathbf{x}) = \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\tilde{\mathbf{x}}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

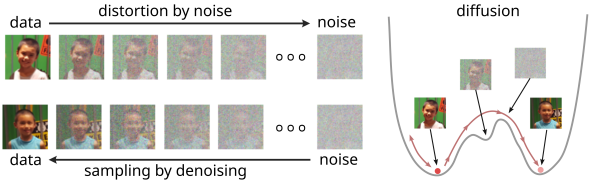
Noise mitigates when density on low-dim manifold.

Multi-level variant of (ii): Use several levels of Gaussian noise $\{\sigma_i\}_{i=1}^L$ and simultaneously estimates scores at all noise levels $s_\theta(\mathbf{x}, \sigma)$, (Song 2019).

Noise Conditional Score Networks (NCSN): $s_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2 \mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Annealed Langevin Dynamics can be used by first sampling for noise level σ_i, $i = 1$ then successively for σ_j, $j > i$, where $\frac{\sigma_i}{\sigma_j} > c > 1$ and $\sigma_L \approx 0$.

Score-Matching Diffusion Methods & SDEs



Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

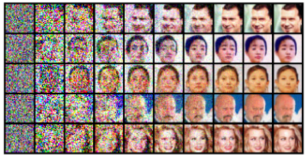
Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$, (ii) denoising score matching using $\tilde{s}(\mathbf{x}) = \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\tilde{\mathbf{x}}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Noise mitigates when density on low-dim manifold.

Annealed Sampling



Song 2020

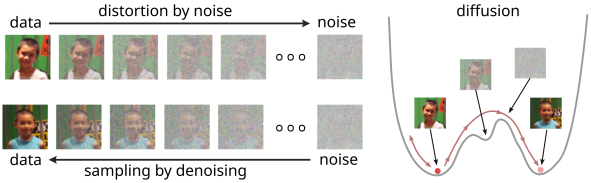
Multi-level variant of (ii): Use several levels of Gaussian noise $\{\sigma_i\}_{i=1}^L$ and simultaneously estimates scores at all noise levels $s_\theta(\mathbf{x}, \sigma)$, (Song 2019).

Noise Conditional Score Networks (NCSN): $s_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2 \mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Annealed Langevin Dynamics can be used by first sampling for noise level σ_i, $i = 1$ then successively for σ_j, $j > i$, where $\frac{\sigma_i}{\sigma_j} > c > 1$ and $\sigma_L \approx 0$.

Diffusive Sampling uses Metropolis-Hastings based on $X_{n+1} = X_n + \frac{\epsilon}{2} s_\theta(X_n, \sigma_{i_n}) + \sqrt{\epsilon}\eta_n$, $\eta_n \sim \mathcal{N}(0, \mathcal{I})$.

Score-Matching Diffusion Methods & SDEs



Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

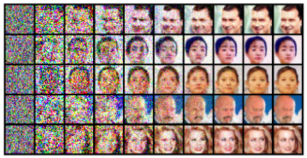
Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$, (ii) denoising score matching using $\tilde{s}(\mathbf{x}) = \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\tilde{\mathbf{x}}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Noise mitigates when density on low-dim manifold.

Annealed Sampling



Song 2020

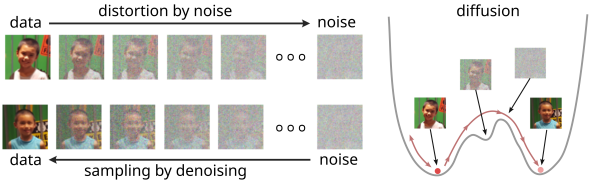
Multi-level variant of (ii): Use several levels of Gaussian noise $\{\sigma_i\}_{i=1}^L$ and simultaneously estimates scores at all noise levels $s_\theta(\mathbf{x}, \sigma)$, (Song 2019).

Noise Conditional Score Networks (NCSN): $s_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2 \mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Annealed Langevin Dynamics can be used by first sampling for noise level σ_i, $i = 1$ then successively for σ_j, $j > i$, where $\frac{\sigma_i}{\sigma_j} > c > 1$ and $\sigma_L \approx 0$.

Diffusive Sampling uses Metropolis-Hastings based on $X_{n+1} = X_n + \frac{\epsilon}{2} s_\theta(X_n, \sigma_{i_n}) + \sqrt{\epsilon}\eta_n$, $\eta_n \sim \mathcal{N}(0, \mathcal{I})$. In limit $n \to \infty$, $\epsilon \to 0$: $\sigma_{i_n} \to \sigma_L$, $X_\infty \sim p_\theta \approx p_{data}$.

Score-Matching Diffusion Methods & SDEs



Score-based methods sample by using Langevin dynamics $dX_t = \frac{1}{2} s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Stationary Distribution: In limit $t \to \infty$, $X_\infty \sim p(\mathbf{x}) \sim \mu_X$.

Objective: minimize $\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

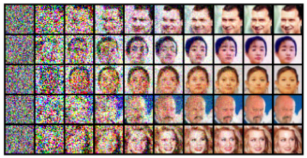
Equivalent (Hyvärinen 2005) to minimizing $\mathbb{E}_{p_{data}}\left[\|\text{tr}(\nabla_x s_\theta(\mathbf{x})) - \frac{1}{2}\|s_\theta(\mathbf{x})\|_2^2\right]$.

$\text{tr}(\nabla_x s_\theta(\mathbf{x}))$ can be computationally expensive.

Strategies: (i) use sliced score matching projecting onto set of random $\mathbf{v}$, (ii) denoising score matching using $\tilde{s}(\mathbf{x}) = \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int q_\sigma(\mathbf{x}|\tilde{\mathbf{x}}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Noise mitigates when density on low-dim manifold.

Annealed Sampling



Song 2020

Multi-level variant of (ii): Use several levels of Gaussian noise $\{\sigma_i\}_{i=1}^L$ and simultaneously estimates scores at all noise levels $s_\theta(\mathbf{x}, \sigma)$, (Song 2019).

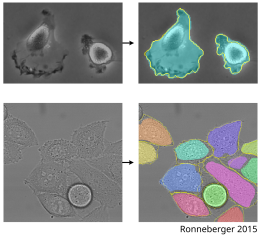
Noise Conditional Score Networks (NCSN): $s_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with $q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2 \mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Annealed Langevin Dynamics can be used by first sampling for noise level σ_i, $i = 1$ then successively for σ_j, $j > i$, where $\frac{\sigma_i}{\sigma_j} > c > 1$ and $\sigma_L \approx 0$.

Diffusive Sampling uses Metropolis-Hastings based on $X_{n+1} = X_n + \frac{\epsilon}{2} s_\theta(X_n, \sigma_{i_n}) + \sqrt{\epsilon}\eta_n$, $\eta_n \sim \mathcal{N}(0, \mathcal{I})$. In limit $n \to \infty$, $\epsilon \to 0$: $\sigma_{i_n} \to \sigma_L$, $X_\infty \sim p_\theta \approx p_{data}$.

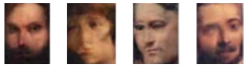
U-Net Architecture

Segmentation Masks



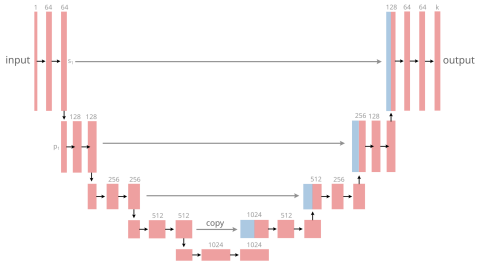
Ronneberger 2015

Image Generation



Naik 2025

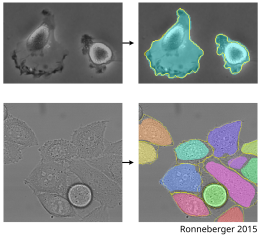
Guo 2024



U-Nets widely used in segmentation tasks and image generation.

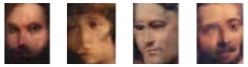
U-Net Architecture

Segmentation Masks



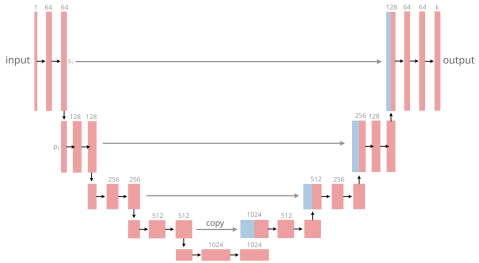
Ronneberger 2015

Image Generation



Naik 2025

Guo 2024

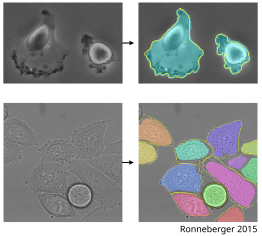


U-Nets widely used in segmentation tasks and image generation.

U-Net Architecture (Ronneberger 2015) uses two stage process for images of size $w \times h$
(i) down-sample for mutli-level feature extraction,
(ii) up-sample for synthesis of image/mask.

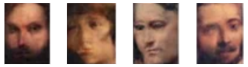
U-Net Architecture

Segmentation Masks



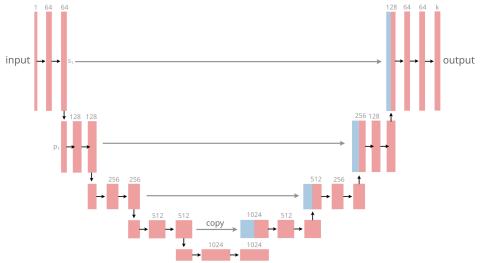
Ronneberger 2015

Image Generation



Naik 2025

Guo 2024



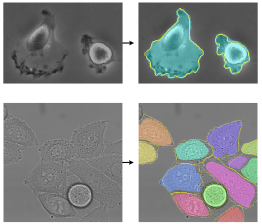
U-Nets widely used in segmentation tasks and image generation.

U-Net Architecture (Ronneberger 2015) uses two stage process for images of size $w \times h$
(i) down-sample for mutli-level feature extraction,
(ii) up-sample for synthesis of image/mask.

Convolutions are used successively to extract context features which are then injected back when upscaling (copied).

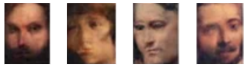
U-Net Architecture

Segmentation Masks



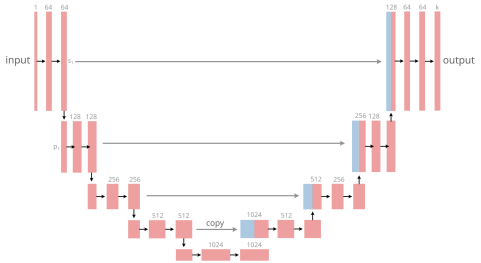
Ronneberger 2015

Image Generation



Naik 2025

Guo 2024



U-Nets widely used in segmentation tasks and image generation.

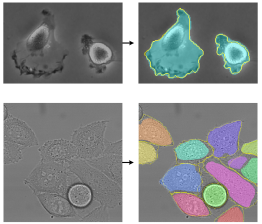
U-Net Architecture (Ronneberger 2015) uses two stage process for images of size $w \times h$
(i) down-sample for mutli-level feature extraction,
(ii) up-sample for synthesis of image/mask.

Convolutions are used successively to extract context features which are then injected back when upscaling (copied).

Pooling operations are used to enhance invariance and reduce $w \times h$ dimensions, such as $\max\text{-pool}$.

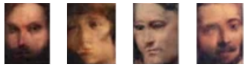
U-Net Architecture

Segmentation Masks



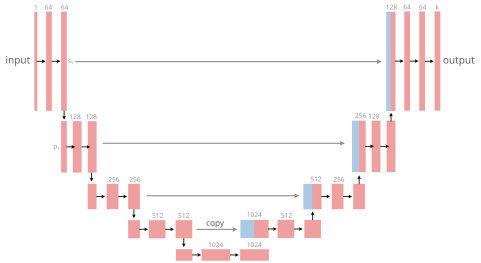
Ronneberger 2015

Image Generation



Naik 2025

Guo 2024



U-Nets widely used in segmentation tasks and image generation.

U-Net Architecture (Ronneberger 2015) uses two stage process for images of size $w \times h$
(i) down-sample for mutli-level feature extraction,
(ii) up-sample for synthesis of image/mask.

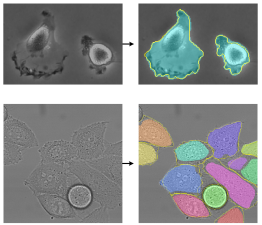
Convolutions are used successively to extract context features which are then injected back when upscaling (copied).

Pooling operations are used to enhance invariance and reduce $w \times h$ dimensions, such as $\mathrm{max\text{-}pool}$.

Each layer produces (s_i, p_i) where s_i denotes the image feature channels and p_i the pooled layer output.

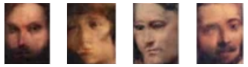
U-Net Architecture

Segmentation Masks



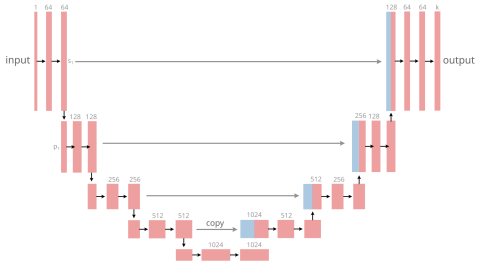
Ronneberger 2015

Image Generation



Naik 2025

Guo 2024



U-Nets widely used in segmentation tasks and image generation.

U-Net Architecture (Ronneberger 2015) uses two stage process for images of size $w \times h$
(i) down-sample for mutli-level feature extraction,
(ii) up-sample for synthesis of image/mask.

Convolutions are used successively to extract context features which are then injected back when upscaling (copied).

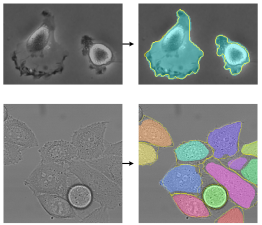
Pooling operations are used to enhance invariance and reduce $w \times h$ dimensions, such as $\mathrm{max\text{-}pool}$.

Each layer produces (s_i, p_i) where s_i denotes the image feature channels and p_i the pooled layer output.

(i) Feature extraction and down-sampling are performed using CNNs on the image to obtain multi-level feature channels.

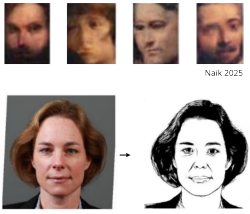
U-Net Architecture

Segmentation Masks



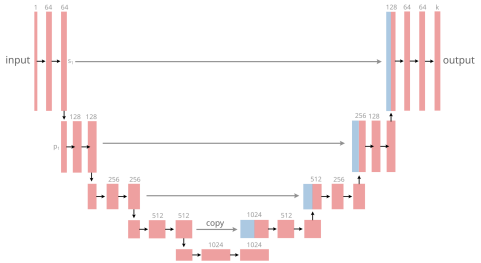
Ronneberger 2015

Image Generation



Naik 2025

Guo 2024



U-Nets widely used in segmentation tasks and image generation.

U-Net Architecture (Ronneberger 2015) uses two stage process for images of size $w \times h$
(i) down-sample for mutli-level feature extraction,
(ii) up-sample for synthesis of image/mask.

Convolutions are used successively to extract context features which are then injected back when upscaling (copied).

Pooling operations are used to enhance invariance and reduce $w \times h$ dimensions, such as $\max\text{-pool}$.

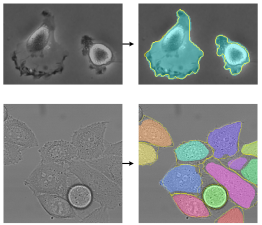
Each layer produces (s_i, p_i) where s_i denotes the image feature channels and p_i the pooled layer output.

(i) Feature extraction and down-sampling are performed using CNNs on the image to obtain multi-level feature channels.

(ii) Up-sampling and feature injection is applied successively to construct a $w \times h \times k$ image or mask having k output feature channels.

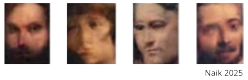
U-Net Architecture

Segmentation Masks



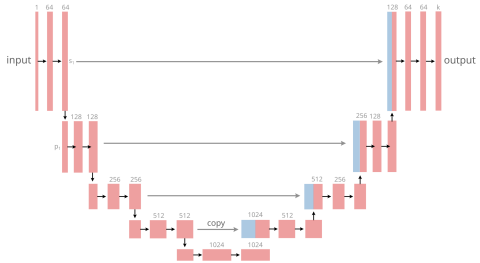
Ronneberger 2015

Image Generation



Naik 2025

Guo 2024



U-Nets widely used in segmentation tasks and image generation.

U-Net Architecture (Ronneberger 2015) uses two stage process for images of size $w \times h$
(i) down-sample for mutli-level feature extraction,
(ii) up-sample for synthesis of image/mask.

Convolutions are used successively to extract context features which are then injected back when upscaling (copied).

Pooling operations are used to enhance invariance and reduce $w \times h$ dimensions, such as $\max\text{-pool}$.

Each layer produces (s_i, p_i) where s_i denotes the image feature channels and p_i the pooled layer output.

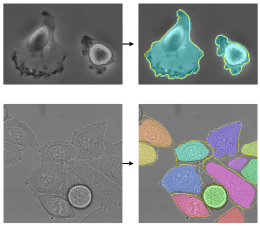
(i) Feature extraction and down-sampling are performed using CNNs on the image to obtain multi-level feature channels.

(ii) Up-sampling and feature injection is applied successively to construct a $w \times h \times k$ image or mask having k output feature channels.

Feature injection is performed by concatenating the previous features for $\tilde{w} \times \tilde{h} \times \tilde{k}$ with the upscaled image features $\tilde{w} \times \tilde{h} \times \tilde{k}'$ to obtain the new image channels $\tilde{w} \times \tilde{h} \times (\tilde{k} + \tilde{k}')$.

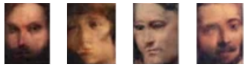
U-Net Architecture

Segmentation Masks



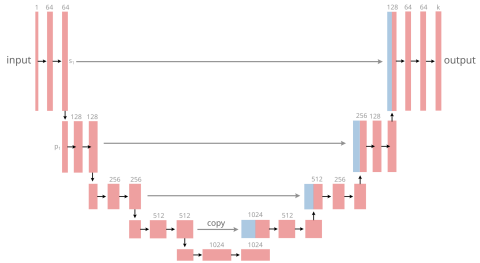
Ronneberger 2015

Image Generation



Naik 2025

Guo 2024



U-Nets widely used in segmentation tasks and image generation.

U-Net Architecture (Ronneberger 2015) uses two stage process for images of size $w \times h$
(i) down-sample for mutli-level feature extraction,
(ii) up-sample for synthesis of image/mask.

Convolutions are used successively to extract context features which are then injected back when upscaling (copied).

Pooling operations are used to enhance invariance and reduce $w \times h$ dimensions, such as $\mathrm{max\text{-}pool}$.

Each layer produces (s_i, p_i) where s_i denotes the image feature channels and p_i the pooled layer output.

(i) Feature extraction and down-sampling are performed using CNNs on the image to obtain multi-level feature channels.

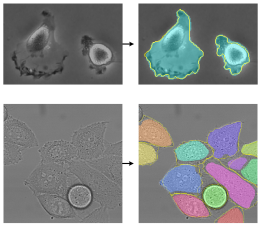
(ii) Up-sampling and feature injection is applied successively to construct a $w \times h \times k$ image or mask having k output feature channels.

Feature injection is performed by concatenating the previous features for $\tilde{w} \times \tilde{h} \times \tilde{k}$ with the upscaled image features $\tilde{w} \times \tilde{h} \times \tilde{k}'$ to obtain the new image channels $\tilde{w} \times \tilde{h} \times (\tilde{k} + \tilde{k}')$.

Injection provides local context information during the upscaling process for constructing the output.

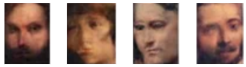
U-Net Architecture

Segmentation Masks



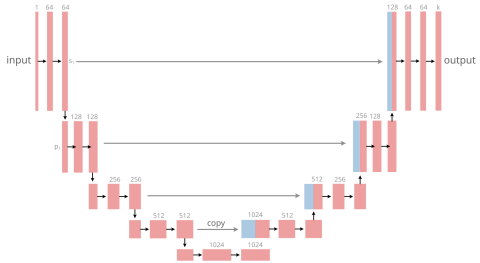
Ronneberger 2015

Image Generation



Naik 2025

Guo 2024



U-Nets widely used in segmentation tasks and image generation.

U-Net Architecture (Ronneberger 2015) uses two stage process for images of size $w \times h$
(i) down-sample for mutli-level feature extraction,
(ii) up-sample for synthesis of image/mask.

Convolutions are used successively to extract context features which are then injected back when upscaling (copied).

Pooling operations are used to enhance invariance and reduce $w \times h$ dimensions, such as $\mathrm{max\text{-}pool}$.

Each layer produces (s_i, p_i) where s_i denotes the image feature channels and p_i the pooled layer output.

(i) Feature extraction and down-sampling are performed using CNNs on the image to obtain multi-level feature channels.

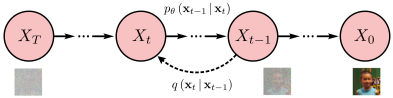
(ii) Up-sampling and feature injection is applied successively to construct a $w \times h \times k$ image or mask having k output feature channels.

Feature injection is performed by concatenating the previous features for $\tilde{w} \times \tilde{h} \times \tilde{k}$ with the upscaled image features $\tilde{w} \times \tilde{h} \times \tilde{k}'$ to obtain the new image channels $\tilde{w} \times \tilde{h} \times (\tilde{k} + \tilde{k}')$.

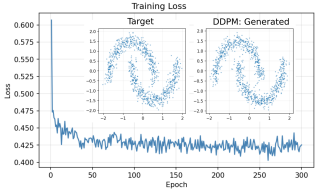
Injection provides local context information during the upscaling process for constructing the output.

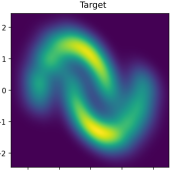
Example: Diffusion Methods (DDPM)

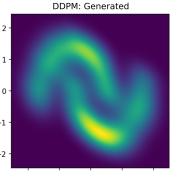
Generative Model $\mathcal{G}_\theta$



Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

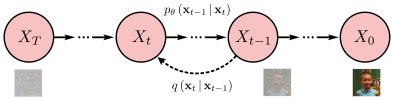


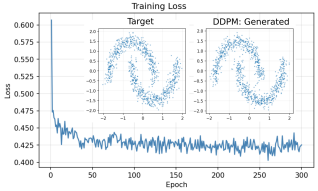


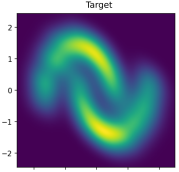


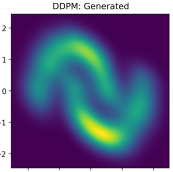
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$







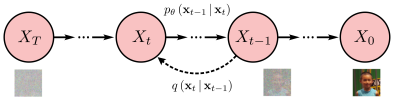


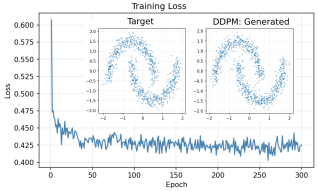
Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

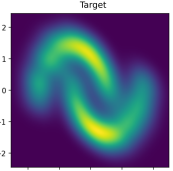
$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

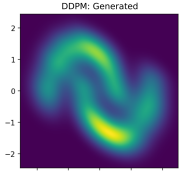
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I}).$

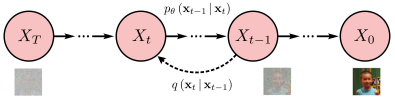
Learning $\mathcal{G}_\theta$:

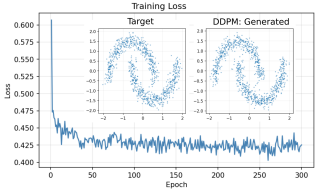
(i) forward-process provides training samples

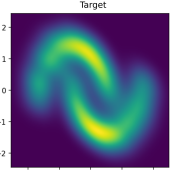
(ii) learned backward denoising process provides sampler.

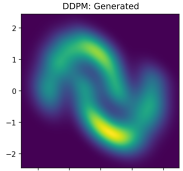
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

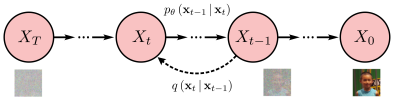
$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

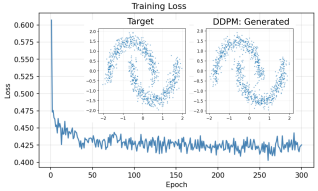
Learning $\mathcal{G}_\theta$**:**
(i) forward-process provides training samples
(ii) learned backward denoising process provides sampler.

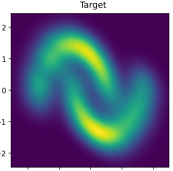
Sampler $\mathcal{G}_\theta$**:** Dffusion process with Markov chain transitions
$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

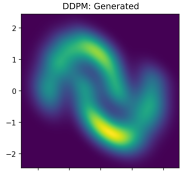
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

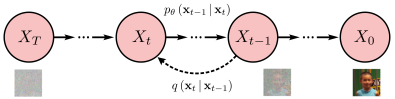
Learning $\mathcal{G}_\theta$:
(i) forward-process provides training samples
(ii) learned backward denoising process provides sampler.

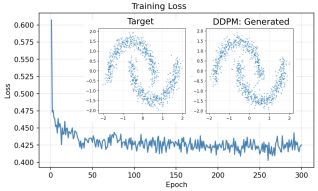
Sampler $\mathcal{G}_\theta$: Dffusion process with Markov chain transitions
$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

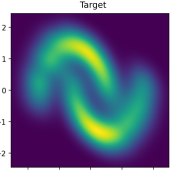
Trained uusing Evidence Lower Bound (ELBO) for log-likelihood $\mathbb{E}[\log(p_\theta(\mathbf{x}_0))] \geq \mathrm{ELBO}(\theta)$.

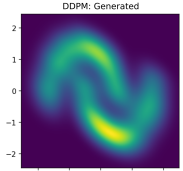
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$:
(i) forward-process provides training samples
(ii) learned backward denoising process provides sampler.

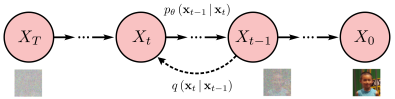
Sampler $\mathcal{G}_\theta$: Dffusion process with Markov chain transitions
$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

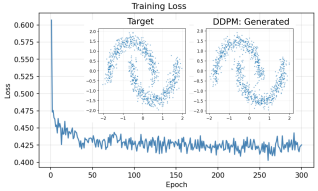
Trained uusing Evidence Lower Bound (ELBO) for log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$.

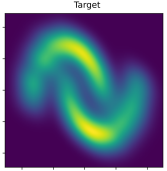
Loss (after simplification):
$\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.

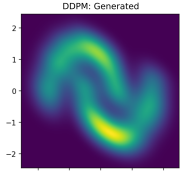
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$:
(i) forward-process provides training samples
(ii) learned backward denoising process provides sampler.

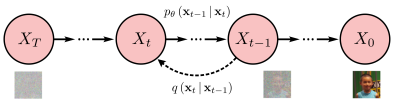
Sampler $\mathcal{G}_\theta$: Dffusion process with Markov chain transitions
$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

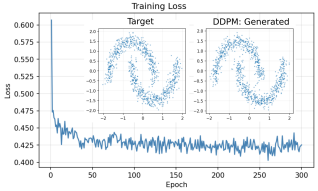
Trained uusing Evidence Lower Bound (ELBO) for log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$.

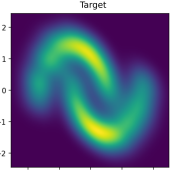
Loss (after simplification):
$\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

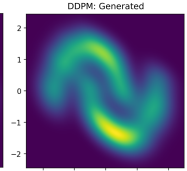
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$:
(i) forward-process provides training samples
(ii) learned backward denoising process provides sampler.

Target Distribution: half-moon dataset

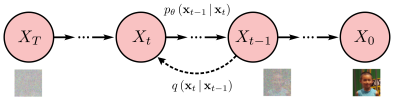
Sampler $\mathcal{G}_\theta$: Dffusion process with Markov chain transitions
$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

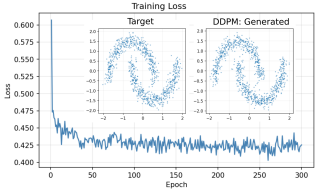
Trained uusing Evidence Lower Bound (ELBO) for log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$.

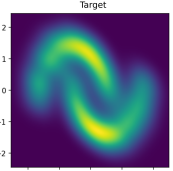
Loss (after simplification):
$\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

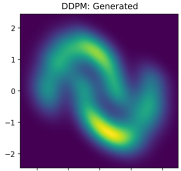
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$:
(i) forward-process provides training samples
(ii) learned backward denoising process provides sampler.

Sampler $\mathcal{G}_\theta$: Dffusion process with Markov chain transitions $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

Trained uusing Evidence Lower Bound (ELBO) for log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$.

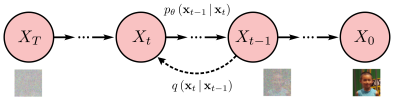
Loss (after simplification):
$\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

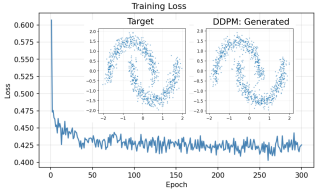
Target Distribution: half-moon dataset

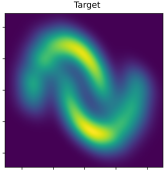
Training DDPM: $N \sim 10^4$ samples, $n_e = 500$ epochs.

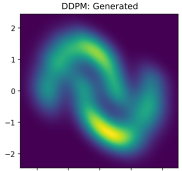
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$:
(i) forward-process provides training samples
(ii) learned backward denoising process provides sampler.

Sampler $\mathcal{G}_\theta$: Dffusion process with Markov chain transitions $p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

Trained uusing Evidence Lower Bound (ELBO) for log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$.

Loss (after simplification):
$\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

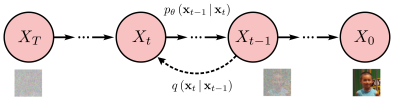
Target Distribution: half-moon dataset

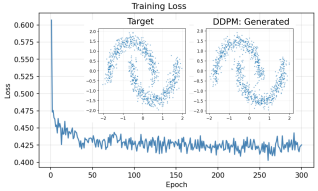
Training DDPM: $N \sim 10^4$ samples, $n_e = 500$ epochs.

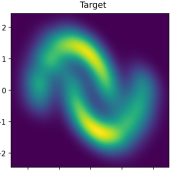
Remarks: Regions separated by low probability states present challenges and act like energy-barriers for the diffusion process.

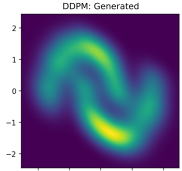
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$:
(i) forward-process provides training samples
(ii) learned backward denoising process provides sampler.

Sampler $\mathcal{G}_\theta$: Dffusion process with Markov chain transitions
$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

Trained uusing Evidence Lower Bound (ELBO) for log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$.

Loss (after simplification):
$\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

Target Distribution: half-moon dataset

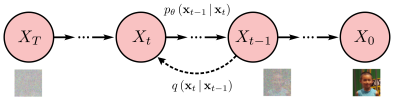
Training DDPM: $N \sim 10^4$ samples, $n_e = 500$ epochs.

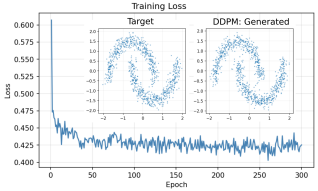
Remarks: Regions separated by low probability states present challenges and act like energy-barriers for the diffusion process.

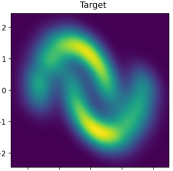
Results: DPPM is able to learn denoising process overcoming these barriers sampling from both regions. Distributions shown above.

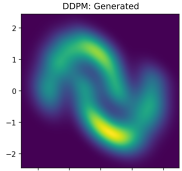
Example: Diffusion Methods (DDPM)

Generative Model $\mathcal{G}_\theta$









Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$:
(i) forward-process provides training samples
(ii) learned backward denoising process provides sampler.

Sampler $\mathcal{G}_\theta$: Dffusion process with Markov chain transitions
$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_\theta(\mathbf{x}_t, t), \boldsymbol{\Sigma}_\theta(\mathbf{x}_t, t))$.

Trained uusing Evidence Lower Bound (ELBO) for log-likelihood $\mathbb{E}\left[\log\left(p_\theta(\mathbf{x}_0)\right)\right] \geq \mathrm{ELBO}(\theta)$.

Loss (after simplification):
$\ell(\theta) = \mathbb{E}_{\mathbf{x}_0, \boldsymbol{\epsilon}}\left[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2\right]$.
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \mathcal{I})$, $\boldsymbol{\epsilon}_\theta(\tilde{\mathbf{x}}, t)$ is learnable Gaussian noise.

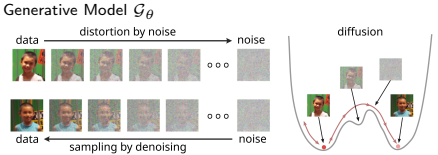
Target Distribution: half-moon dataset

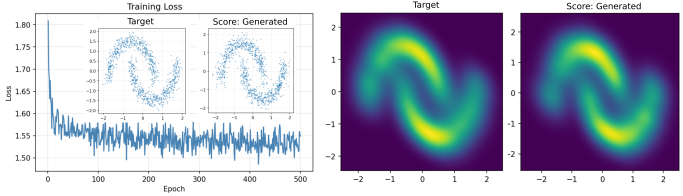
Training DDPM: $N \sim 10^4$ samples, $n_e = 500$ epochs.

Remarks: Regions separated by low probability states present challenges and act like energy-barriers for the diffusion process.

Results: DPPM is able to learn denoising process overcoming these barriers sampling from both regions. Distributions shown above.

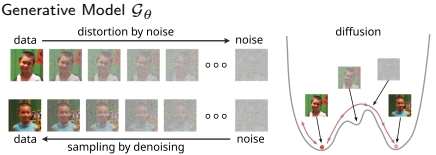
Example: Diffusion Methods (Score Matching)

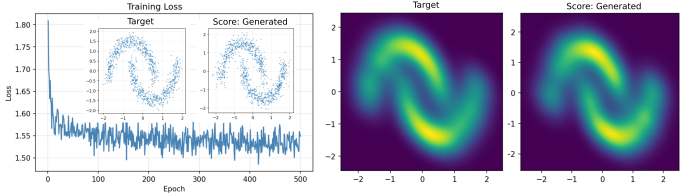




Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

Example: Diffusion Methods (Score Matching)

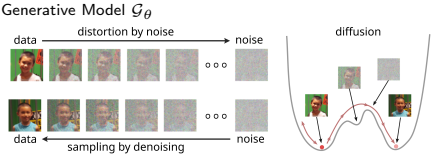


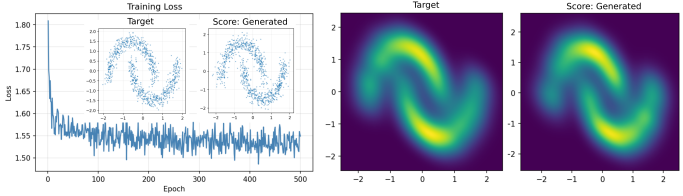


Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Example: Diffusion Methods (Score Matching)



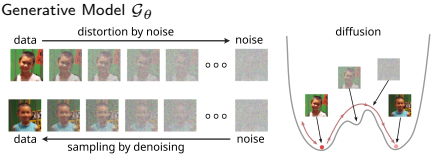


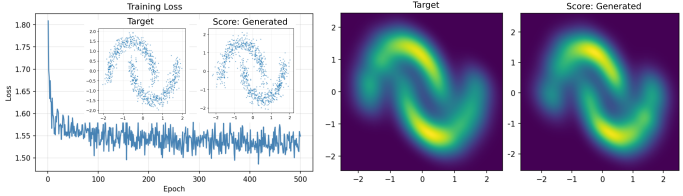
Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

Example: Diffusion Methods (Score Matching)





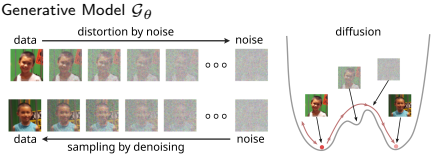
Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

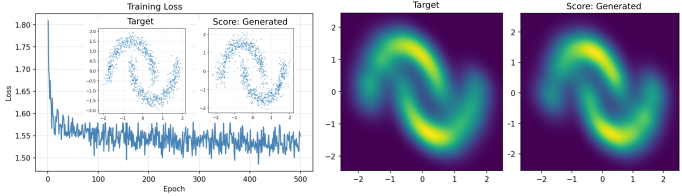
$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

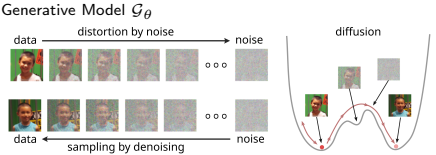
Learning $\mathcal{G}_\theta$**:**(approximate objective)

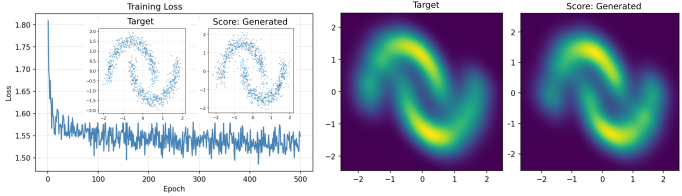
$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

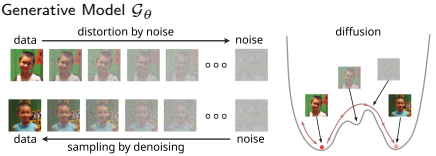
Learning $\mathcal{G}_\theta$**:**(approximate objective)

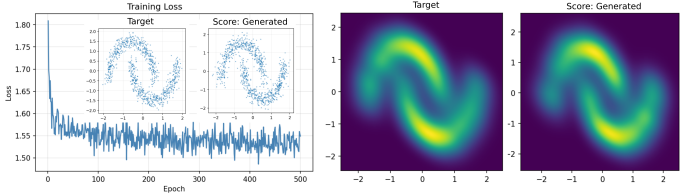
$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

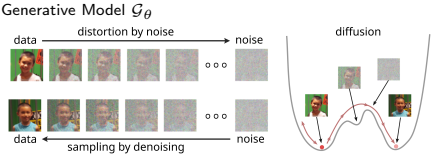
Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

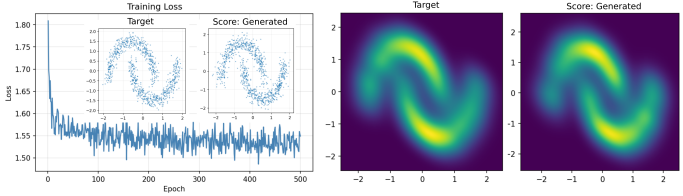
$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Loss: (noise level σ)

$\ell(\theta, \sigma) = \mathbb{E}_{p_{data},\sigma}\left[\|\text{tr}(\nabla_x \tilde{s}_\theta(\mathbf{x}, \sigma)) - \frac{1}{2}\|\tilde{s}_\theta(\mathbf{x}, \sigma)\|_2^2\right]$.

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

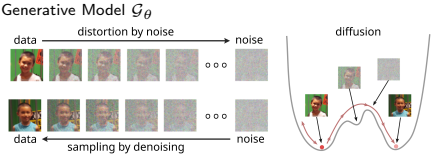
$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

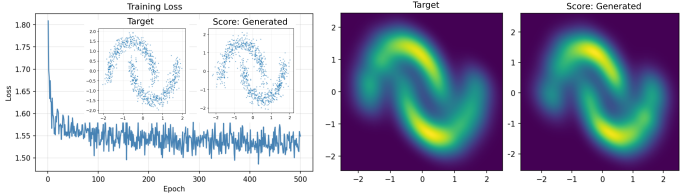
Loss: (noise level σ)

$\ell(\theta, \sigma) = \mathbb{E}_{p_{data},\sigma}\left[\|\text{tr}(\nabla_x \tilde{s}_\theta(\mathbf{x}, \sigma)) - \frac{1}{2}\|\tilde{s}_\theta(\mathbf{x}, \sigma)\|_2^2\right]$.

Noise Conditional Score Networks (NCSN):

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Loss: (noise level σ)

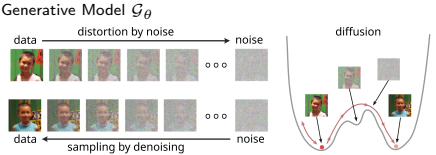
$\ell(\theta, \sigma) = \mathbb{E}_{p_{data},\sigma}\left[\|\text{tr}(\nabla_x \tilde{s}_\theta(\mathbf{x}, \sigma)) - \frac{1}{2}\|\tilde{s}_\theta(\mathbf{x}, \sigma)\|_2^2\right]$.

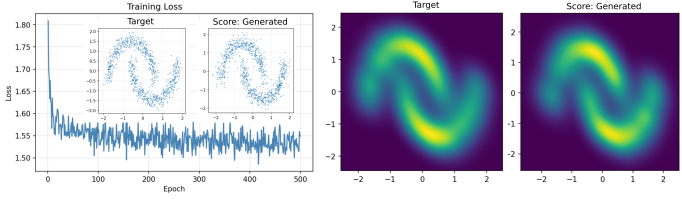
Noise Conditional Score Networks (NCSN):

$\tilde{s}_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with

$q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2\mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

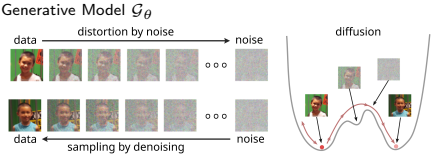
Loss: (noise level σ)

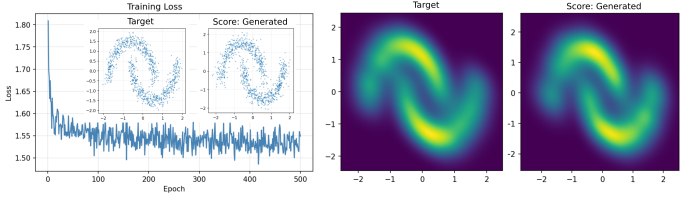
$\ell(\theta, \sigma) = \mathbb{E}_{p_{data},\sigma}\left[\|\text{tr}(\nabla_x \tilde{s}_\theta(\mathbf{x}, \sigma)) - \frac{1}{2}\|\tilde{s}_\theta(\mathbf{x}, \sigma)\|_2^2\right]$.

Noise Conditional Score Networks (NCSN):
$\tilde{s}_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with
$q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2\mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Target Distribution: half-moon dataset

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Loss: (noise level σ)

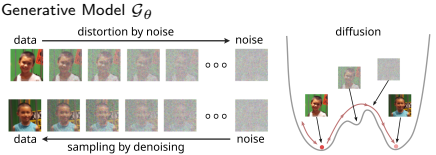
$\ell(\theta, \sigma) = \mathbb{E}_{p_{data},\sigma}\left[\|\text{tr}(\nabla_x \tilde{s}_\theta(\mathbf{x}, \sigma)) - \frac{1}{2}\|\tilde{s}_\theta(\mathbf{x}, \sigma)\|_2^2\right]$.

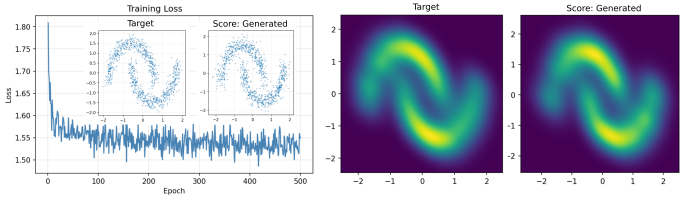
Noise Conditional Score Networks (NCSN):
$\tilde{s}_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with
$q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2\mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

Target Distribution: half-moon dataset

Training NCSN: $N \sim 10^4$ samples, $n_e = 500$ epochs.

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Loss: (noise level σ)

$\ell(\theta, \sigma) = \mathbb{E}_{p_{data},\sigma}\left[\|\text{tr}(\nabla_x \tilde{s}_\theta(\mathbf{x}, \sigma)) - \frac{1}{2}\|\tilde{s}_\theta(\mathbf{x}, \sigma)\|_2^2\right]$.

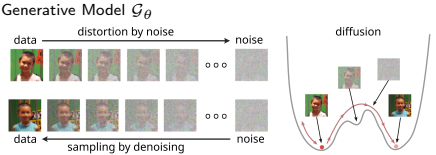
Noise Conditional Score Networks (NCSN):
$\tilde{s}_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with
$q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2\mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

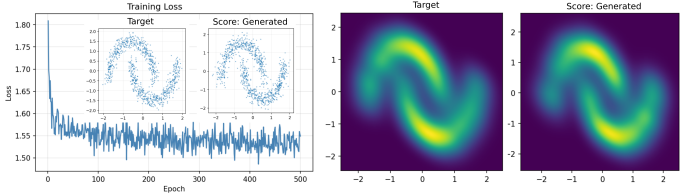
Target Distribution: half-moon dataset

Training NCSN: $N \sim 10^4$ samples, $n_e = 500$ epochs.

Remarks: Low probability regions can act like energy-barriers for the diffusion process. NCSN uses multiple noise levels similar to annealed sampling.

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Loss: (noise level σ)

$\ell(\theta, \sigma) = \mathbb{E}_{p_{data},\sigma}\left[\|\text{tr}(\nabla_x \tilde{s}_\theta(\mathbf{x}, \sigma)) - \frac{1}{2}\|\tilde{s}_\theta(\mathbf{x}, \sigma)\|_2^2\right]$.

Noise Conditional Score Networks (NCSN):
$\tilde{s}_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with
$q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2\mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

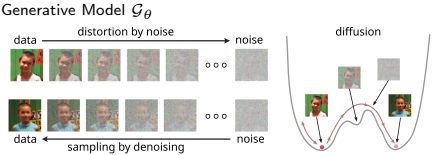
Target Distribution: half-moon dataset

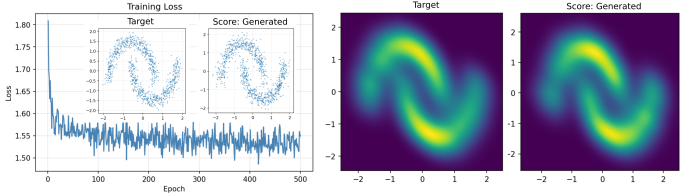
Training NCSN: $N \sim 10^4$ samples, $n_e = 500$ epochs.

Remarks: Low probability regions can act like energy-barriers for the diffusion process. NCSN uses multiple noise levels similar to annealed sampling.

Results: NCSN exhibits good sampling of the disconnected regions. Distributions shown above.

Example: Diffusion Methods (Score Matching)





Task: Learn $\mathcal{G}_\theta$ for generating samples $\tilde{\mathbf{X}}$ for the target distribution $p(\mathbf{x})$.

$\tilde{\mathbf{X}} = \mathcal{G}_\theta(\mathbf{Z}) \sim p(\mathbf{x}), \quad \mathbf{Z} \sim \mathcal{N}(0, \mathcal{I})$.

Learning $\mathcal{G}_\theta$**:**(approximate objective)

$\frac{1}{2}\mathbb{E}_{p_{data}}\left[\|s_\theta(\mathbf{x}) - \nabla_x \log(p_{data}(\mathbf{x}))\|_2^2\right]$.

Sampler $\mathcal{G}_\theta$**:** Dffusion process with Langevin SDE

$dX_t = \frac{1}{2}s_\theta(X_t)dt + dW_t$, with $s_\theta = \nabla_x \log(p_\theta(\mathbf{x}))$ (score).

Loss: (noise level σ)

$\ell(\theta, \sigma) = \mathbb{E}_{p_{data},\sigma}\left[\|\text{tr}(\nabla_x \tilde{s}_\theta(\mathbf{x}, \sigma)) - \frac{1}{2}\|\tilde{s}_\theta(\mathbf{x}, \sigma)\|_2^2\right]$.

Noise Conditional Score Networks (NCSN):
$\tilde{s}_\theta(\mathbf{x}, \sigma) \approx \nabla_x \log(q_\sigma(\mathbf{x}))$ with
$q_\sigma(\mathbf{x}) = \int \mathcal{N}(\tilde{\mathbf{x}}; \mathbf{x}, \sigma^2\mathcal{I}) p_{data}(\tilde{\mathbf{x}}) d\tilde{\mathbf{x}}$.

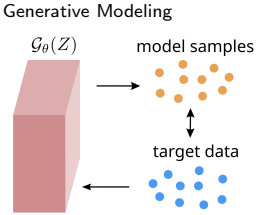
Target Distribution: half-moon dataset

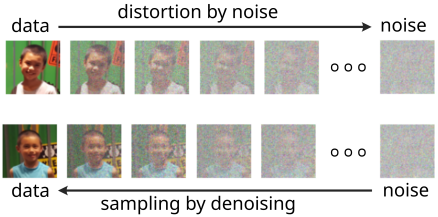
Training NCSN: $N \sim 10^4$ samples, $n_e = 500$ epochs.

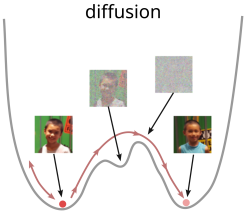
Remarks: Low probability regions can act like energy-barriers for the diffusion process. NCSN uses multiple noise levels similar to annealed sampling.

Results: NCSN exhibits good sampling of the disconnected regions. Distributions shown above.

Conclusions

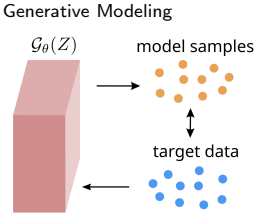


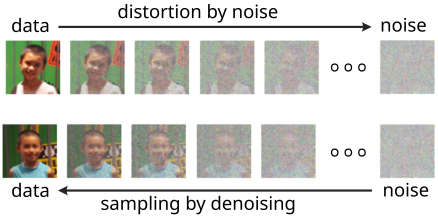


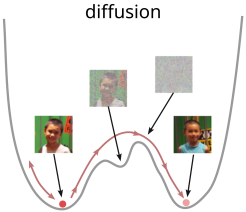


Diffusion Methods for a target distribution μ_X learn implicit generative models $\mathcal{G}_\theta$. This provides samples $\tilde{X} = \mathcal{G}_\theta(Z) \sim \mu_X$ generated from noise Z.

Conclusions



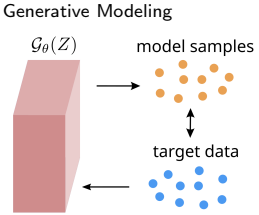


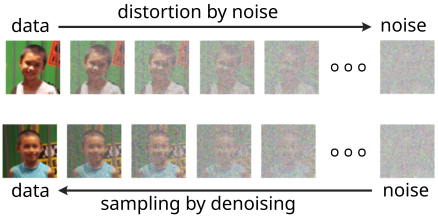


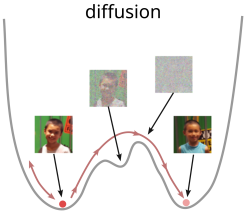
Diffusion Methods for a target distribution μ_X learn implicit generative models $\mathcal{G}_\theta$. This provides samples $\tilde{X} = \mathcal{G}_\theta(Z) \sim \mu_X$ generated from noise Z.

High-dimensional probability distributions have been shown to be effectively sampled with these methods.

Conclusions





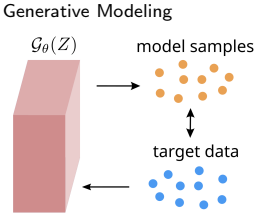


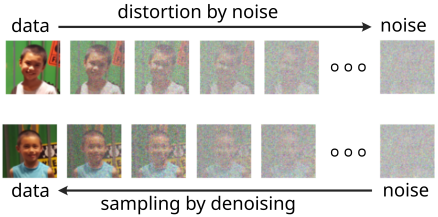
Diffusion Methods for a target distribution μ_X learn implicit generative models $\mathcal{G}_\theta$. This provides samples $\tilde{X} = \mathcal{G}_\theta(Z) \sim \mu_X$ generated from noise Z.

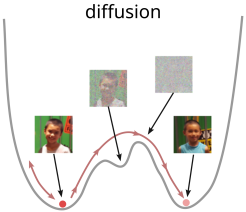
High-dimensional probability distributions have been shown to be effectively sampled with these methods.

Used for generating images, audio, and even video sequences. Many other applications.

Conclusions







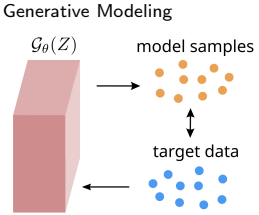
Diffusion Methods for a target distribution μ_X learn implicit generative models $\mathcal{G}_\theta$. This provides samples $\tilde{X} = \mathcal{G}_\theta(Z) \sim \mu_X$ generated from noise Z.

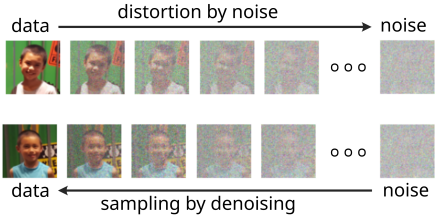
High-dimensional probability distributions have been shown to be effectively sampled with these methods.

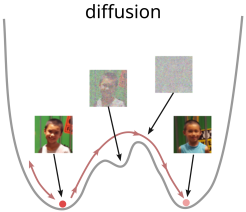
Used for generating images, audio, and even video sequences. Many other applications.

Multi-level noise methods provide efficient scalable sampling approaches (similar to simulated annealing).

Conclusions







Diffusion Methods for a target distribution μ_X learn implicit generative models $\mathcal{G}_\theta$. This provides samples $\tilde{X} = \mathcal{G}_\theta(Z) \sim \mu_X$ generated from noise Z.

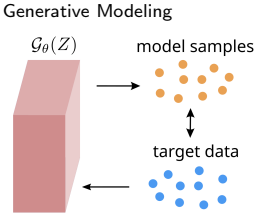
High-dimensional probability distributions have been shown to be effectively sampled with these methods.

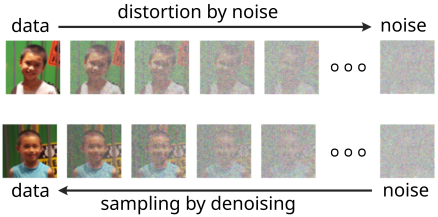
Used for generating images, audio, and even video sequences. Many other applications.

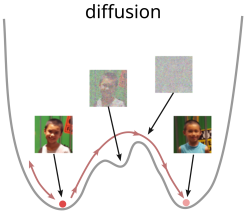
Multi-level noise methods provide efficient scalable sampling approaches (similar to simulated annealing).

Generative AI Services: Scalability has had a big impact resulting in the current era of AI.

Conclusions







Diffusion Methods for a target distribution μ_X learn implicit generative models $\mathcal{G}_\theta$. This provides samples $\tilde{X} = \mathcal{G}_\theta(Z) \sim \mu_X$ generated from noise Z.

High-dimensional probability distributions have been shown to be effectively sampled with these methods.

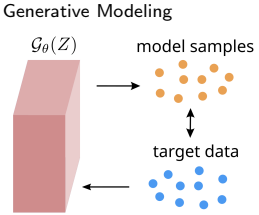
Used for generating images, audio, and even video sequences. Many other applications.

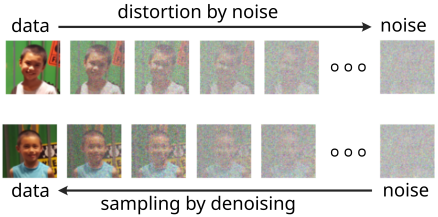
Multi-level noise methods provide efficient scalable sampling approaches (similar to simulated annealing).

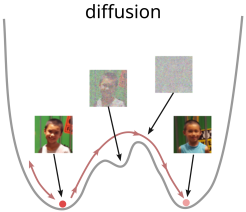
Generative AI Services: Scalability has had a big impact resulting in the current era of AI.

Crucial components in Midjourney, DALL-E, Google Gemini, and other AI services.

Conclusions







Diffusion Methods for a target distribution μ_X learn implicit generative models $\mathcal{G}_\theta$. This provides samples $\tilde{X} = \mathcal{G}_\theta(Z) \sim \mu_X$ generated from noise Z.

High-dimensional probability distributions have been shown to be effectively sampled with these methods.

Used for generating images, audio, and even video sequences. Many other applications.

Multi-level noise methods provide efficient scalable sampling approaches (similar to simulated annealing).

Generative AI Services: Scalability has had a big impact resulting in the current era of AI.

Crucial components in Midjourney, DALL-E, Google Gemini, and other AI services.