Evaluating and tuning precision and recall for Generative Models

Data Science Lab

Alexandre Vérine

Junior AI Fellow of University PSL,
Centre des Données, ENS-PSL

22 October 2026



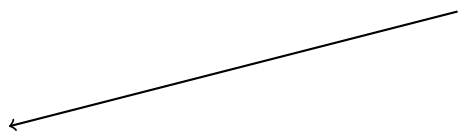




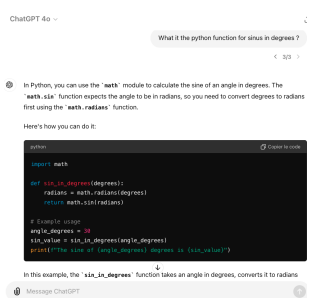
Generative Models

Context

In the past few years, generative models have made significant progress in various domains of application.



Text Generation

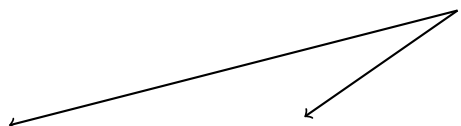


eg: Code Completion

Generative Models

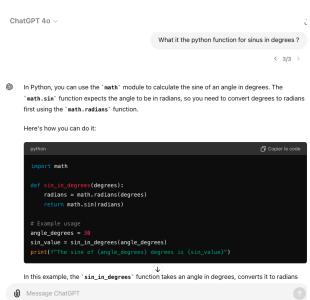
Context

In the past few years, generative models have made significant progress in various domains of application.

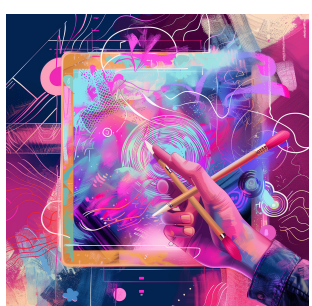


Text Generation

Image/Video Generation



eg: Code Completion

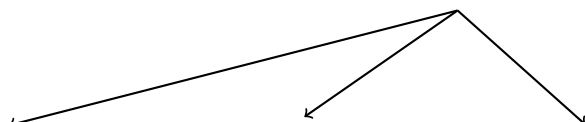


eg: Media Industry

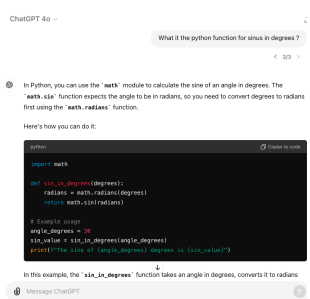
Generative Models

Context

In the past few years, generative models have made significant progress in various domains of application.



Text Generation | Image/Video Generation | Sound Generation



eg: Code Completion



eg: Media Industry

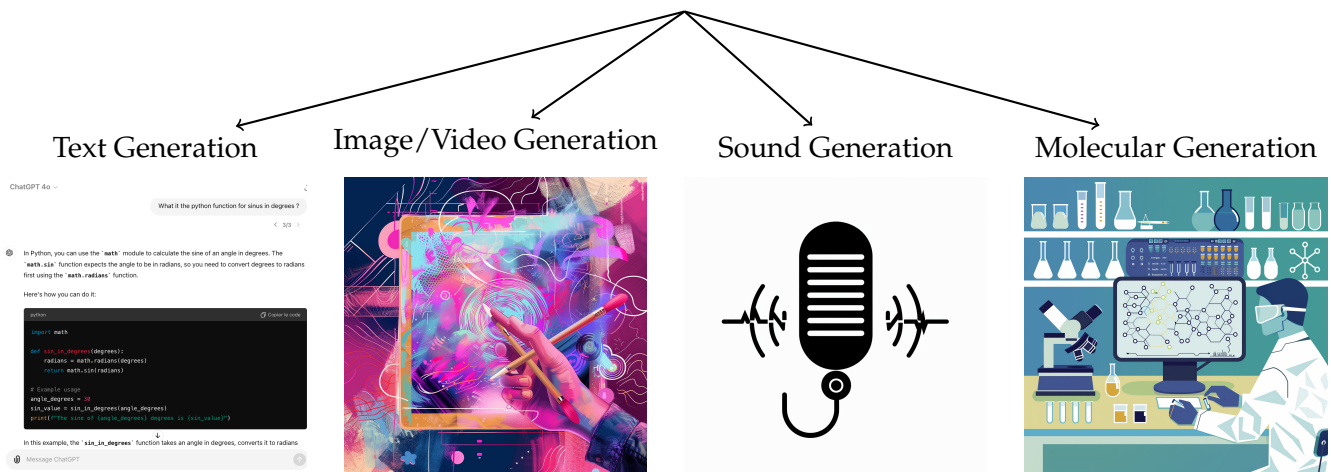


eg: Speech Synthesis

Generative Models

Context

In the past few years, generative models have made significant progress in various domains of application.



The various performances of Generative Models

Motivation

As the generation becomes better, the evaluation becomes more challenging.

DALL·E 2 (2023)



?

Midjourney v5 (2023)



Prompt: *A dog playing with a child.*

The various performances of Generative Models

Motivation

As the generation becomes better, the evaluation becomes more challenging.

DALL·E 2 (2023)



$\neq$

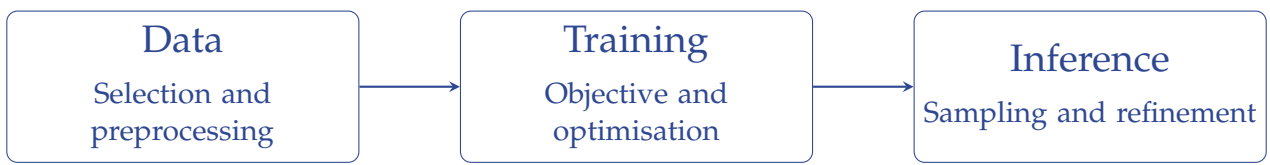
Midjourney v5 (2023)



Prompt: *A dog playing with a child.*

Where can we change the trade-off?

Three levers around the same generative model



Architecture and capacity shape the problem.

In A2, the generator architecture stays fixed; these three levers remain open.

The various performances of Generative Models

Traditional Metrics

Two metrics are often used to evaluate generative models:

- Inception Score (IS) [23]
- Fréchet Inception Distance (FID) [15]

Question : Can they separately evaluate quality and diversity of generative models?

Assume that we have a pre-trained classifier Inception-v3:

Definition 1.1 (Inception Score)

Let denote $\mathbb{P}(Y|\boldsymbol{x})$ be the conditional class distribution of an image $\boldsymbol{x}$ given by the Inception-v3 model and $\mathbb{P}(Y)$ the class distribution in dataset sampled from P. The Inception Score is defined as:

$$\mathrm{IS}(\widehat{P}) = \exp\left(\mathbb{E}_{\boldsymbol{x}\sim\widehat{P}}\left[\mathcal{D}_{\mathrm{KL}}(\mathbb{P}(Y|\boldsymbol{x})\|\mathbb{P}(Y))\right]\right)$$

where $\mathcal{D}_{\mathrm{KL}}$ is the Kullback-Leibler divergence.

With H being the entropy function, IS can be reformulated as:

$$\log\left(\mathrm{IS}(\widehat{P})\right) = H\left(\mathbb{E}_{\boldsymbol{x}\sim\widehat{P}}\left[\mathbb{P}(Y|\boldsymbol{x})\right]\right) - \mathbb{E}_{\boldsymbol{x}\sim\widehat{P}}\left[H\left(\mathbb{P}(Y|\boldsymbol{x})\right)\right].$$

Traditional Metrics

Inception Score

$$\log\left(\mathrm{IS}(\widehat{P})\right) = H\left(\mathbb{E}_{\boldsymbol{x}\sim\widehat{P}}\left[\mathbb{P}(Y|\boldsymbol{x})\right]\right) - \mathbb{E}_{\boldsymbol{x}\sim\widehat{P}}\left[H\left(\mathbb{P}(Y|\boldsymbol{x})\right)\right].$$

Inception Score is maximized :

- if for every $\boldsymbol{x} \sim \widehat{P}$, $H_Y\left(\mathbb{P}(Y|\boldsymbol{x})\right)$ is minimized. This suggests that the classification model is highly confident in predicting a singular label per image, which implies that the images are clearly recognizable and of high quality.
- if the entropy of $\mathbb{E}_{\boldsymbol{x}\sim\widehat{P}}\left[\mathbb{P}(Y|\boldsymbol{x})\right]$ is maximized. The label predictions must be uniformly distributed over all possible labels. This indicates that the generative model generates heterogeneous labels, thereby ensuring diversity.

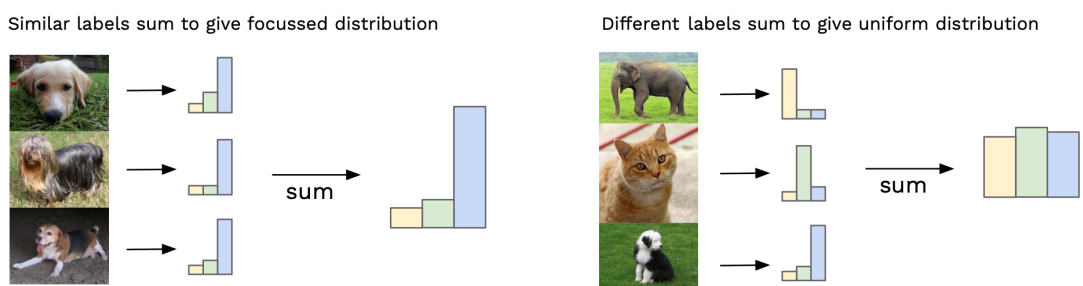


Figure. Source: medium.com

Traditional Metrics

IS falls short

A confident classifier is not an evaluation of the data distribution

One good image per class can score well, even with almost no diversity within each class.

- **Confidence is not realism:** distorted images can fool the classifier.
- **The labels matter:** ImageNet classes may not describe your dataset.
- **No comparison with real samples:** even the real distribution need not maximise IS.

[11, 6, 3, 22]

Traditional Metrics

Fréchet Inception Distance

The Fréchet Inception Distance (FID) compares the statistics of generated samples to real samples based on the features extracted using the output of the last pooling layer of Inception-v3, which is in dimension 2048, we denote it $\phi : \mathbb{R}^d \to \mathbb{R}^m$. We take two sets of samples $\{\boldsymbol{x}_1^{\text{real}}, \dots, \boldsymbol{x}_N^{\text{real}}\}$ drawn from both P and $\{\boldsymbol{x}_1^{\text{fake}}, \dots, \boldsymbol{x}_N^{\text{fake}}\}$ drawn from $\widehat{P}$. We consider the empirical mean and covariance of the latent representation of the samples:

$$\boldsymbol{\mu} = \frac{1}{N}\sum_{i=1}^{N} \phi(\boldsymbol{x}_i^{\text{real}}) \quad \text{and} \quad \boldsymbol{\Sigma} = \frac{1}{N-1}\sum_{i=1}^{N} \left(\phi(\boldsymbol{x}_i^{\text{real}}) - \boldsymbol{\mu}\right)\left(\phi(\boldsymbol{x}_i^{\text{real}}) - \boldsymbol{\mu}\right)^\top,$$

$$\hat{\boldsymbol{\mu}} = \frac{1}{N}\sum_{i=1}^{N} \phi(\boldsymbol{x}_i^{\text{fake}}) \quad \text{and} \quad \widehat{\boldsymbol{\Sigma}} = \frac{1}{N-1}\sum_{i=1}^{N} \left(\phi(\boldsymbol{x}_i^{\text{fake}}) - \hat{\boldsymbol{\mu}}\right)\left(\phi(\boldsymbol{x}_i^{\text{fake}}) - \hat{\boldsymbol{\mu}}\right)^\top$$

Definition 1.2 (Fréchet Inception Distance)

The Fréchet Inception Distance is defined as the Wasserstein-2 distance between two multivariate Gaussians $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $\mathcal{N}(\hat{\boldsymbol{\mu}}, \widehat{\boldsymbol{\Sigma}})$:

$$\text{FID} = \|\boldsymbol{\mu} - \hat{\boldsymbol{\mu}}\|^2 + \text{Tr}\left(\boldsymbol{\Sigma} + \widehat{\boldsymbol{\Sigma}} - 2(\boldsymbol{\Sigma}\widehat{\boldsymbol{\Sigma}})^{1/2}\right)$$

One number mixes quality and diversity

A lower FID does not tell you *what* improved, or what was lost.

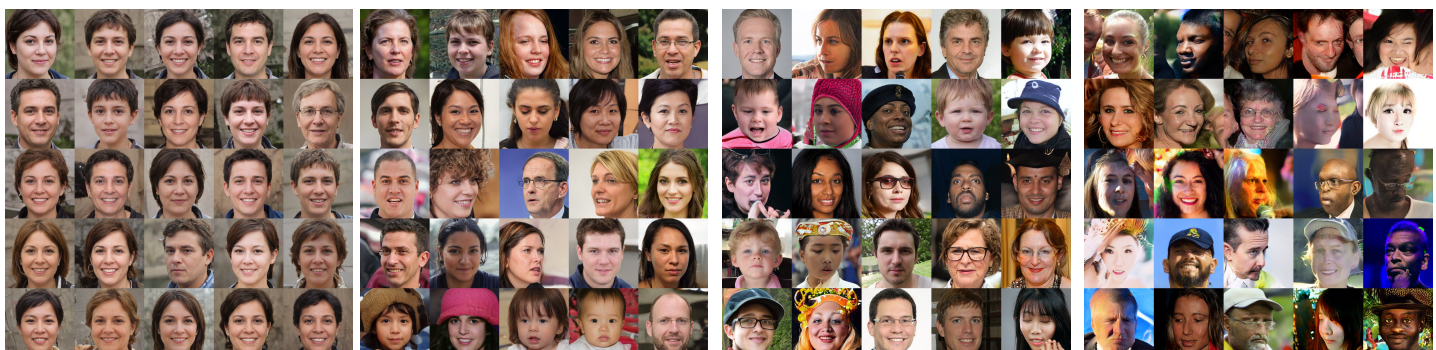
- **Only two moments:** mean and covariance miss finer distributional differences.
- **A Gaussian approximation:** feature distributions can be multimodal.
- **Feature-dependent:** Inception features may miss details that matter to us.

[15, 6, 10]

Traditional Metrics

FID falls short

For example, Kynkäänniemi et al. [18] highlights the limitations of FID with samples drawn from StyleGAN:



(a) Set A - FID= 91.7 (b) Set B - FID= 16.9 (c) Set C - FID= 4.5 (d) Set D - FID= 16.7

Outline

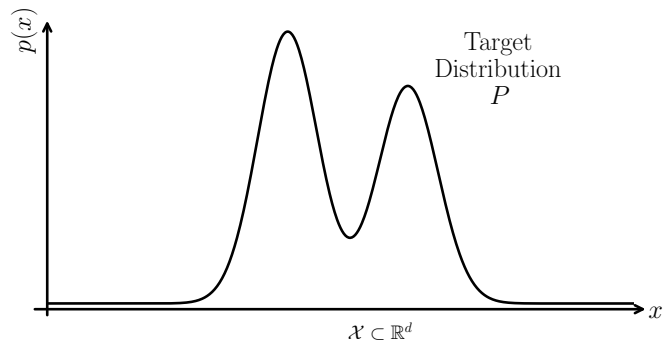
In this presentation, we discuss on evaluating, optimizing and improving quality and diversity of generative models:

1. **Evaluating:** How can we assess quality and diversity independently in Generative Models?
2. **Tuning:** Can we optimize a specific trade-off between quality and diversity?
3. **Improving:** How can we improve the quality and diversity of a pre-trained generative models?

Evaluating:
How can we assess quality and diversity independtly in Generative Models?

Generative Models

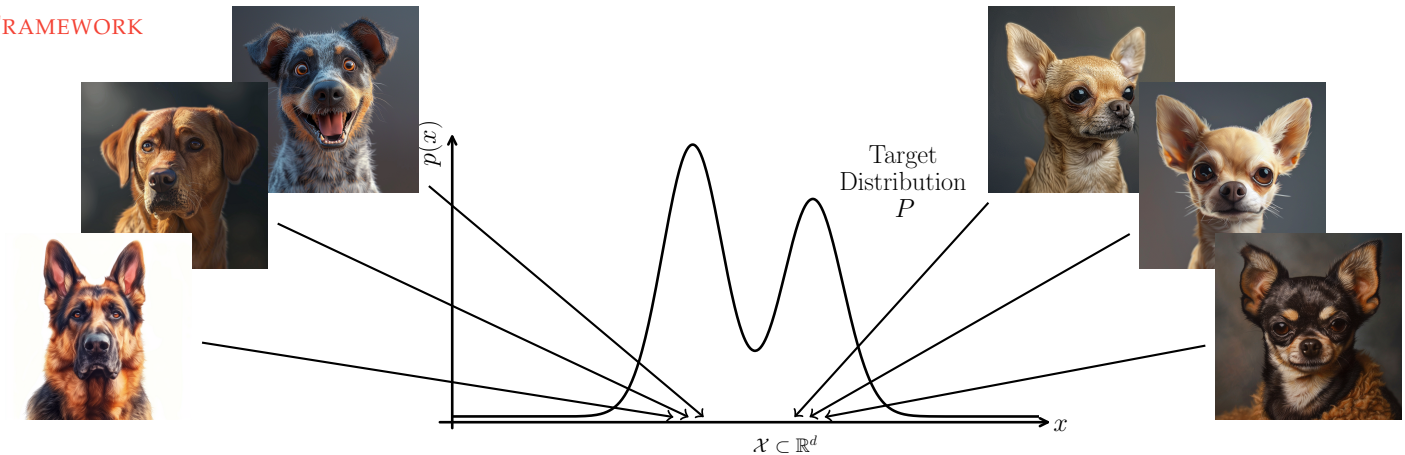
Framework



- Assumption: There is an unknown *target distribution* P in $\mathcal{X} \subset \mathbb{R}^d$.

Generative Models

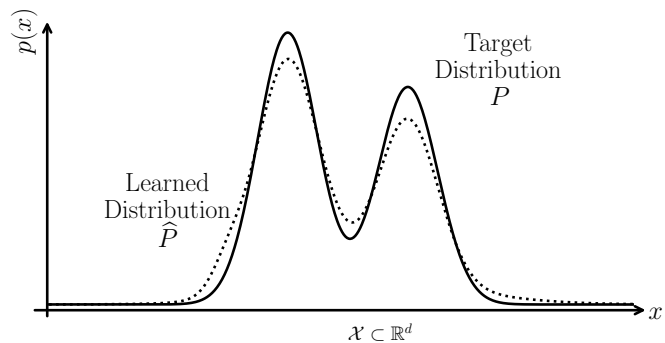
Framework



- Assumption: There is an unknown *target distribution* P in $\mathcal{X} \subset \mathbb{R}^d$.

Generative Models

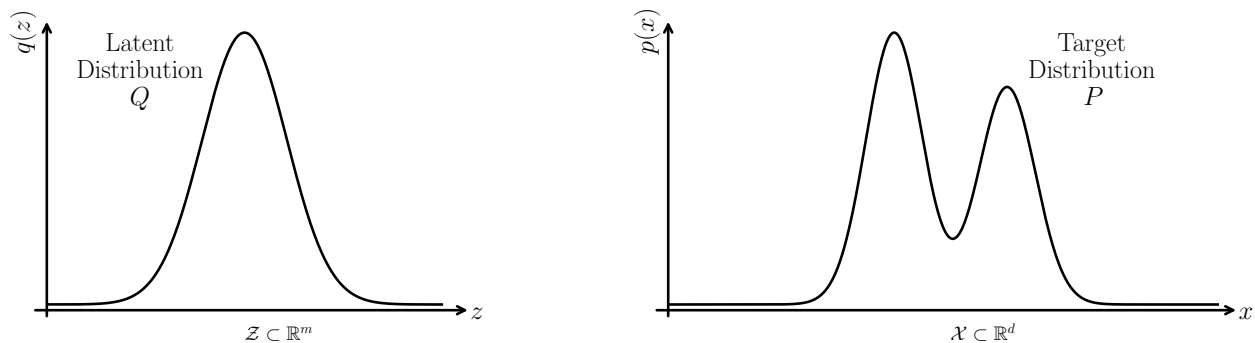
Framework



- Assumption: There is an unknown *target distribution* P in $\mathcal{X} \subset \mathbb{R}^d$.
- Goal: Learn a *parameterized distribution* $\widehat{P}$ that approximate P:

Generative Models

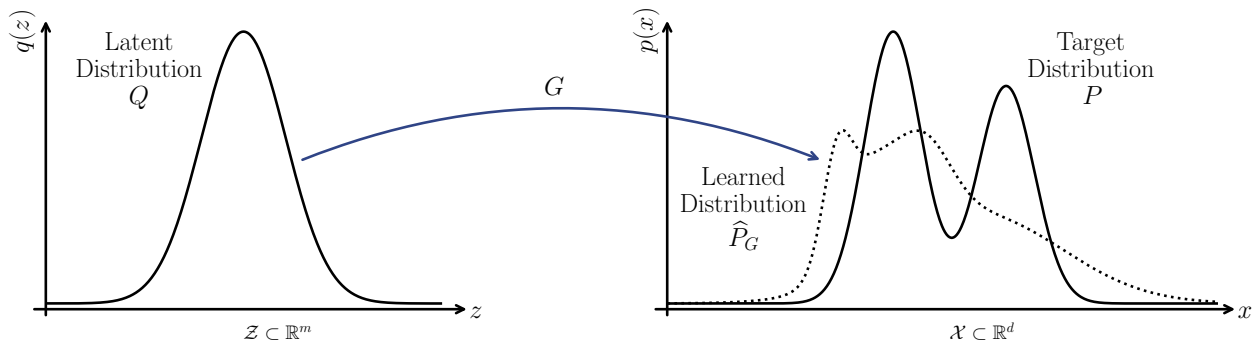
Framework



- Assumption: There is an unknown *target distribution* P in $\mathcal{X} \subset \mathbb{R}^d$.
- Goal: Learn a *parameterized distribution* $\widehat{P}$ that approximate P:
  1. Consider a distribution Q in *a latent space* $\mathcal{X} \subset \mathbb{R}^m$, usually $\mathcal{N}(0, I_m)$.

Generative Models

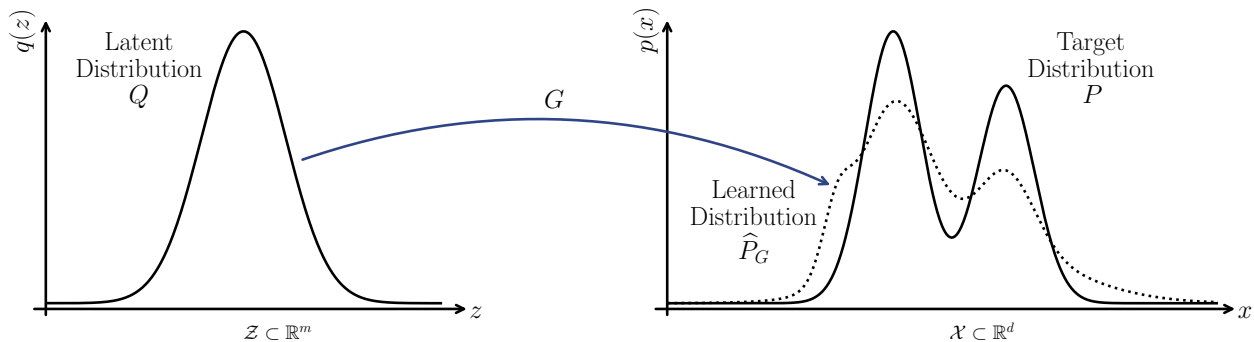
Framework



- Assumption: There is an unknown *target distribution* P in $\mathcal{X} \subset \mathbb{R}^d$.
- Goal: Learn a *parameterized distribution* $\widehat{P}_G$ that approximate P:
  1. Consider a distribution Q in *a latent space* $\mathcal{X} \subset \mathbb{R}^m$, usually $\mathcal{N}(0, I_m)$.
  2. Take *a generator model* G represented by a neural network. Take $\widehat{P}_G = G\#Q$.

Generative Models

Framework

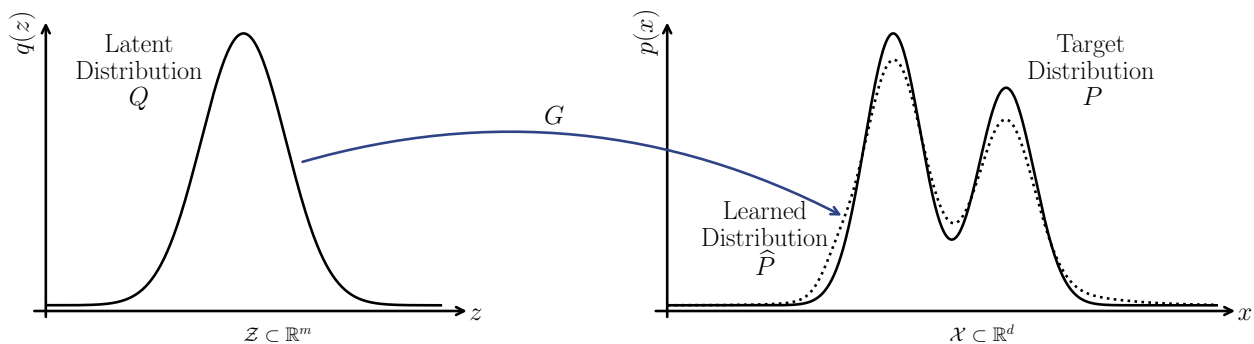


- Assumption: There is an unknown *target distribution* P in $\mathcal{X} \subset \mathbb{R}^d$.
- Goal: Learn a *parameterized distribution* $\widehat{P}_G$ that approximate P:
  1. Consider a distribution Q in *a latent space* $\mathcal{X} \subset \mathbb{R}^m$, usually $\mathcal{N}(0, I_m)$.
  2. Take *a generator model* G represented by a neural network. Take $\widehat{P}_G = G\#Q$.
  3. Compute G^{opt} that minimize *a dissimilarity measure* D between P and $\widehat{P}_G$:

$$G^{\text{opt}} = \underset{G}{\operatorname{argmin}}\, D(P, \widehat{P}_G)$$

Generative Models

Framework



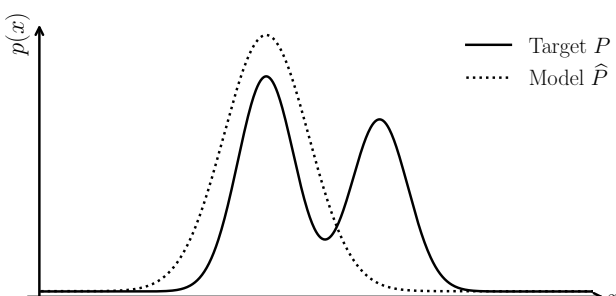
- Assumption: There is an unknown *target distribution* P in $\mathcal{X} \subset \mathbb{R}^d$.
- Goal: Learn a *parameterized distribution* $\widehat{P}$ that approximate P:
  1. Consider a distribution Q in *a latent space* $\mathcal{X} \subset \mathbb{R}^m$, usually $\mathcal{N}(0, I_m)$.
  2. Take *a generator model* G represented by a neural network. Take $\widehat{P} = G\#Q$.
  3. Compute G^{opt} that minimize *a dissimilarity measure* D between P and $\widehat{P}$:

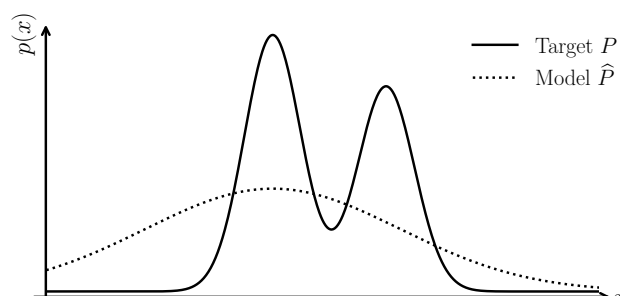
$$G^{\text{opt}} = \underset{G}{\operatorname{argmin}}\, D(P, \widehat{P})$$

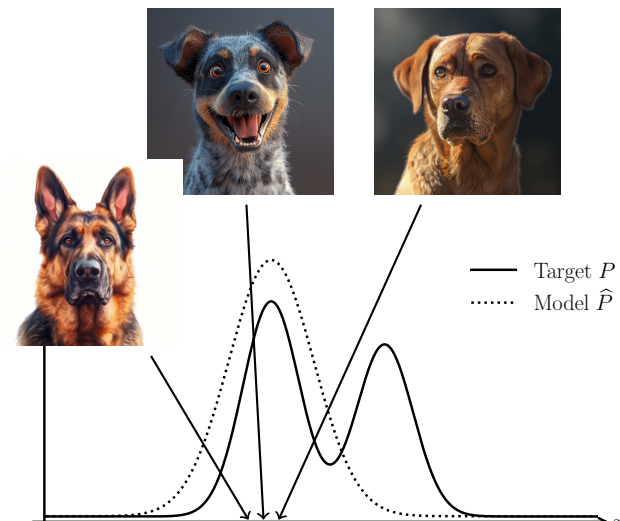
Generative Models

In Practice

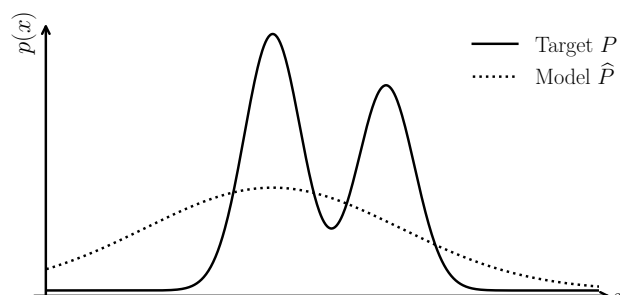
$$\widehat{P} \neq P$$



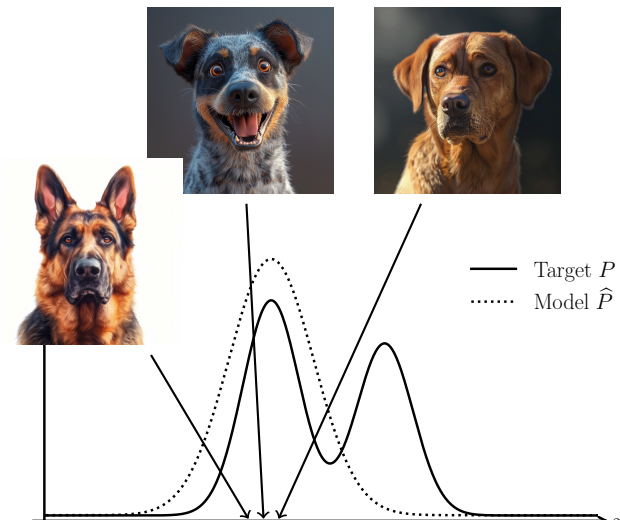




$\widehat{P} \neq P$

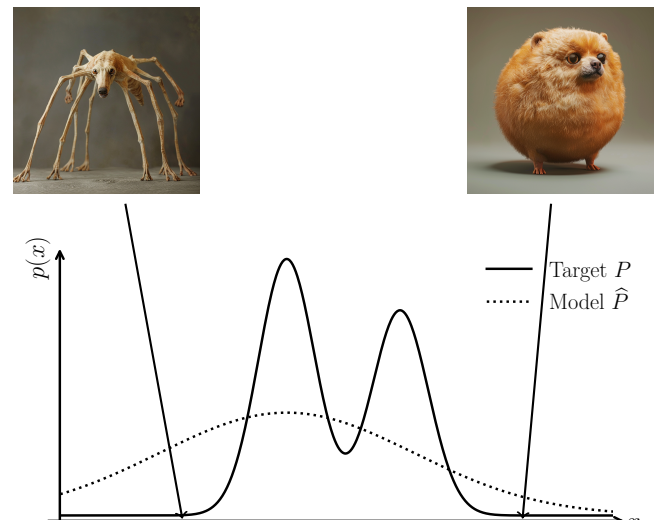


Low Diversity



Low Diversity

$\widehat{P} \neq P$



Low Quality

Precision and Recall for Generative Models

Metrics to evaluate quality and diversity

To assess models, we use the notion of Precision and Recall, inspired from Information Retrieval:

Quality

Diversity

To assess models, we use the notion of Precision and Recall, inspired from Information Retrieval:

Quality

$\downarrow$

Precision

$=$

What proportion of generated samples are realistic?

Diversity

Precision and Recall for Generative Models

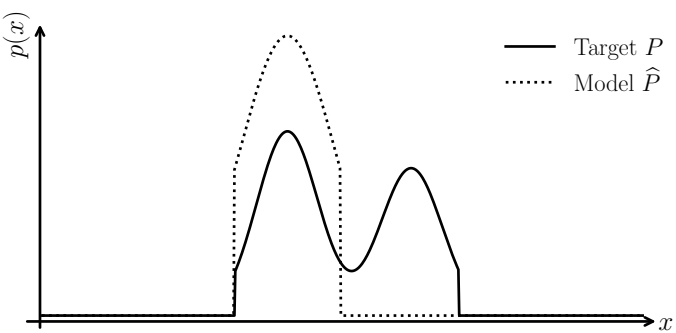
Metrics to evaluate quality and diversity

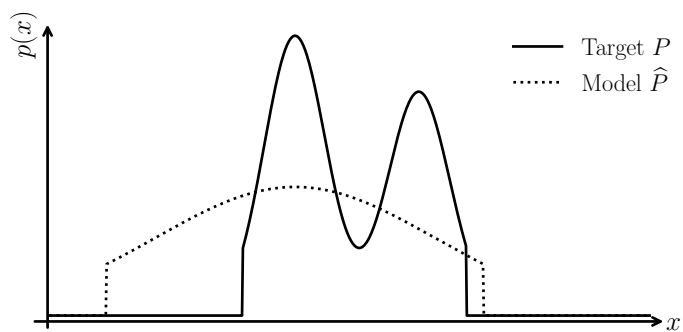
To assess models, we use the notion of Precision and Recall, inspired from Information Retrieval:

Quality	Diversity
$\downarrow$	$\downarrow$
Precision	Recall
$=$	$=$
What proportion of generated samples are realistic?	What proportion of real samples can be generated?

Precision and Recall for Generative Models

For finite support





Precision and Recall for Generative Models

For finite support

Definition 1.3 (Support-Based Precision and Recall - [18].)

For any distributions $P \in \mathcal{P}(\mathcal{X})$ and $\widehat{P} \in \mathcal{P}(\mathcal{X})$, we say that the distribution P has precision $\bar{\alpha}$ at recall $\bar{\beta}$ with respect to $\widehat{P}$ if

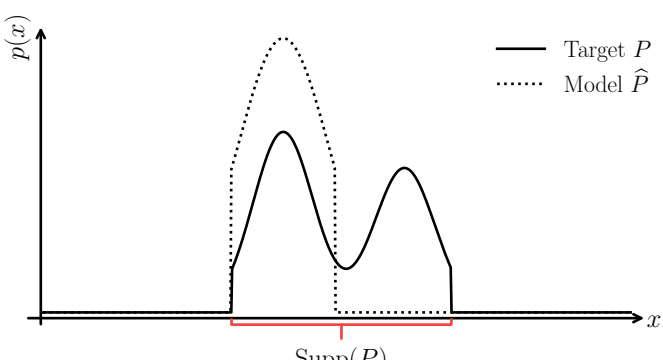
$$\bar{\alpha} \coloneqq \widehat{P}(\mathrm{Supp}(P)) \quad \textit{and} \quad \bar{\beta} \coloneqq P(\mathrm{Supp}(\widehat{P})).$$

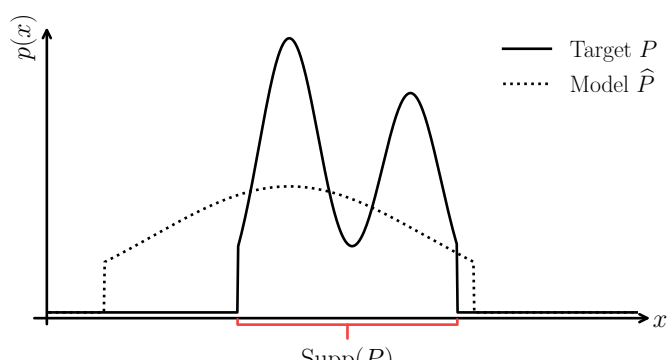
Precision and Recall for Generative Models

For finite support

Precision for finite support is the proportion of generated data that lies on the support of the real data:

$$\bar{\alpha} = \widehat{P}(\text{Supp}(P)).$$



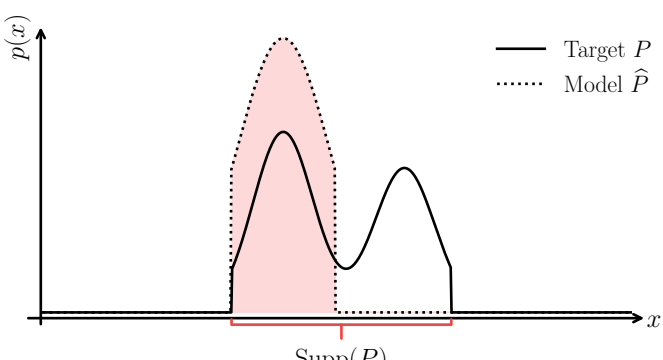


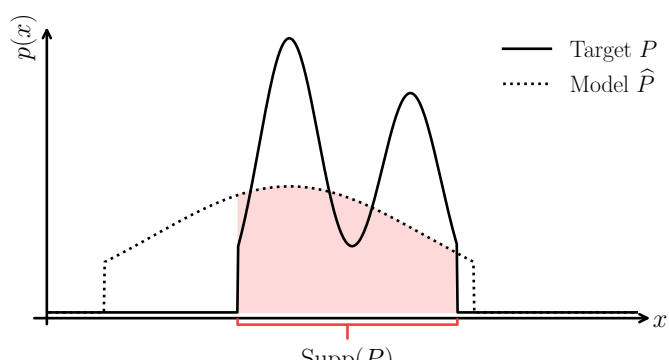
Precision and Recall for Generative Models

For finite support

Precision for finite support is the proportion of generated data that lies on the support of the real data:

$$\bar{\alpha} = \widehat{P}(\text{Supp}(P)).$$



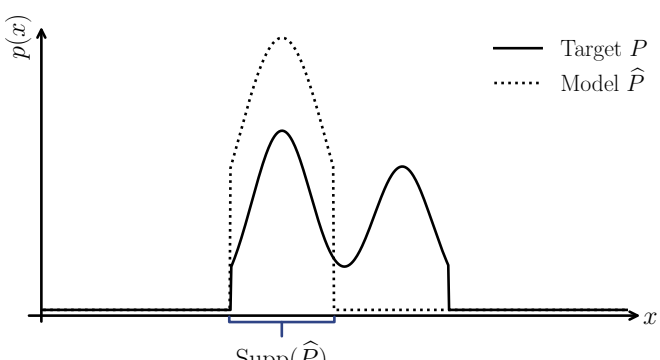


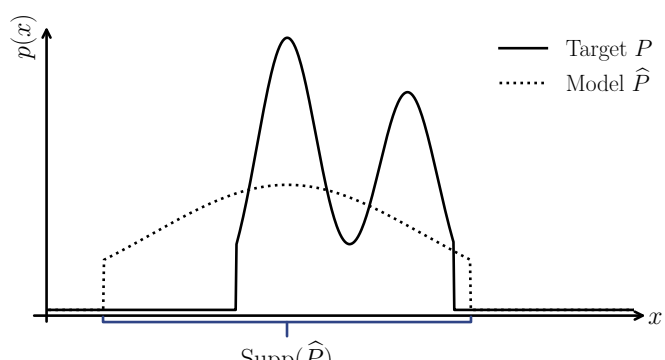
Precision and Recall for Generative Models

For finite support

Recall for finite support is the proportion of the support of the real data that is covered by the generated data:

$$\bar{\beta} = P(\text{Supp}(\widehat{P})).$$



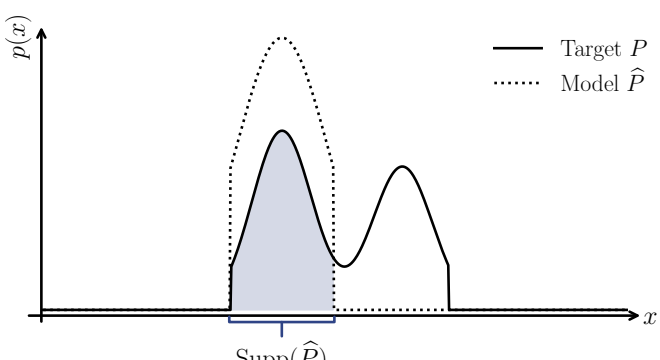


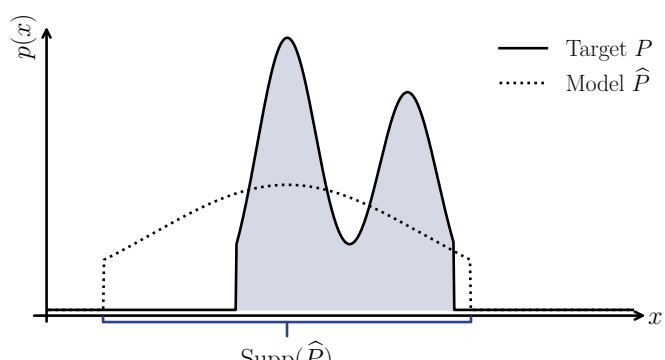
Precision and Recall for Generative Models

For finite support

Recall for finite support is the proportion of the support of the real data that is covered by the generated data:

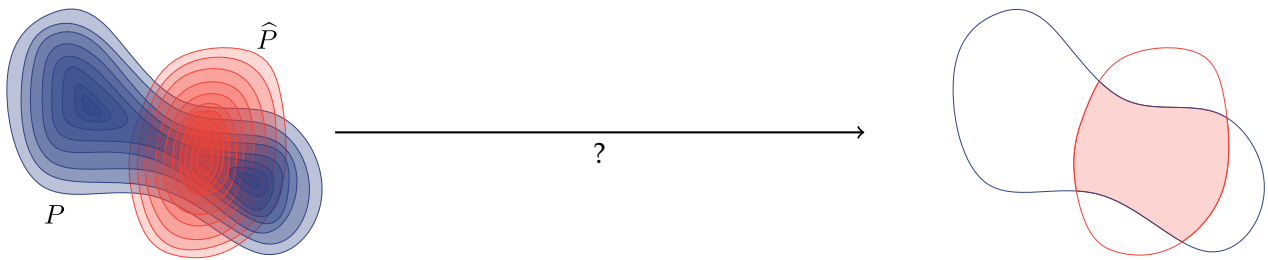
$$\bar{\beta} = P(\text{Supp}(\widehat{P})).$$





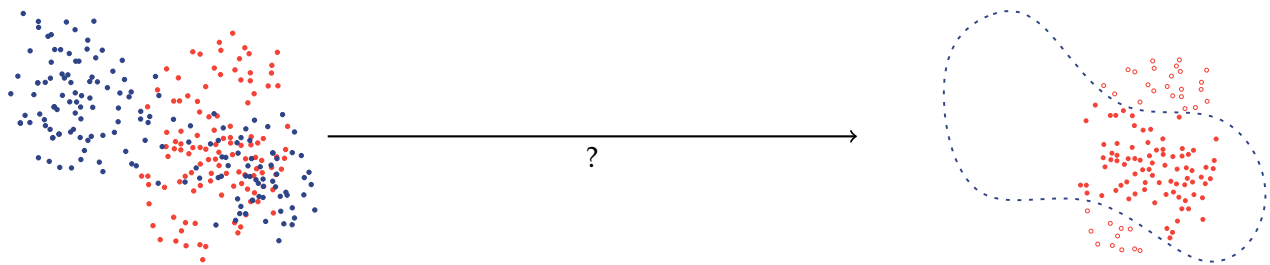
Precision and Recall for Generative Models

Estimating the Precision and Recall



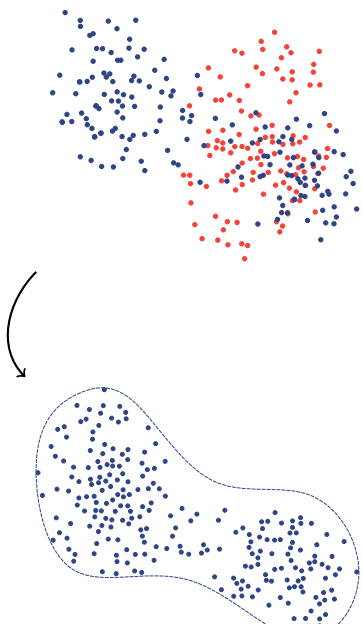
Precision and Recall for Generative Models

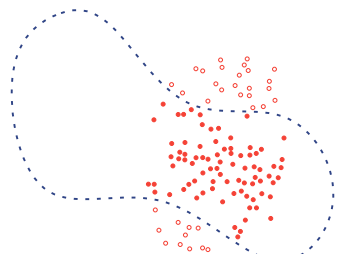
Estimating the Precision and Recall



Precision and Recall for Generative Models

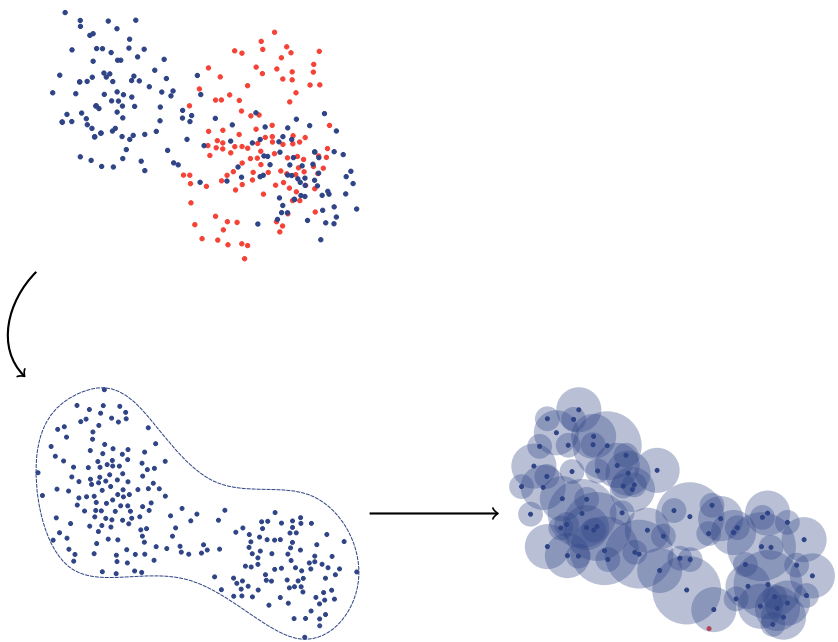
Estimating the Precision and Recall

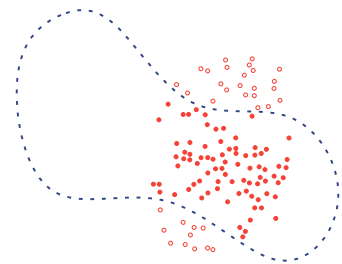




Precision and Recall for Generative Models

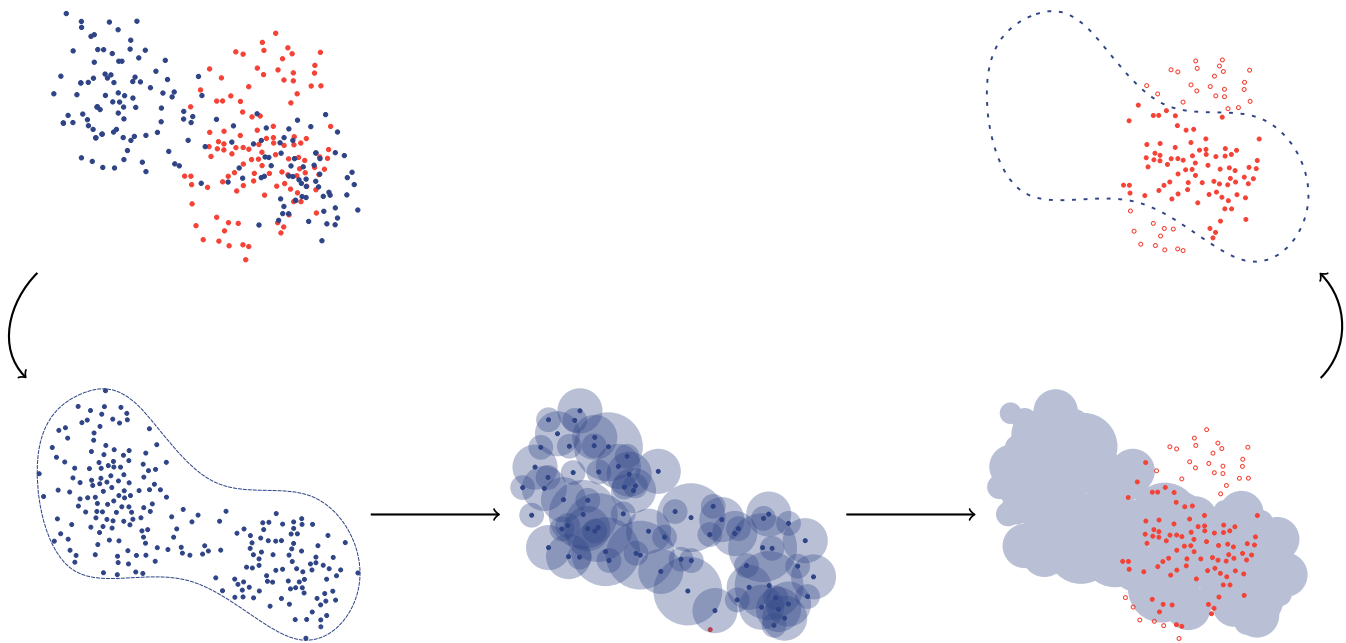
Estimating the Precision and Recall





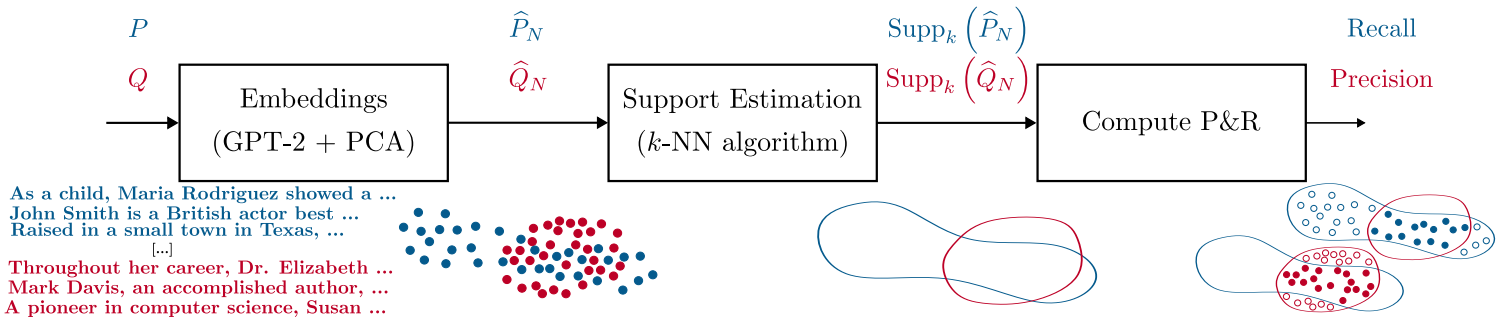
Precision and Recall for Generative Models

Estimating the Precision and Recall



Precision and Recall for Generative Models

For LLMs



Precision and Recall for Generative Models

In Practice

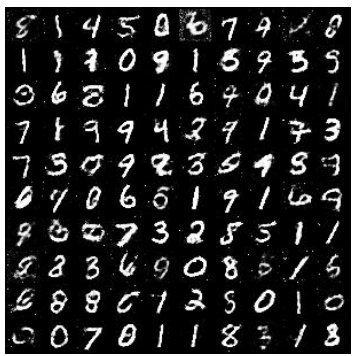
MNIST Dataset [29]



Precision and Recall for Generative Models

In Practice

Low Precision High Recall



Precision: 0.54 Recall: 0.91

MNIST Dataset [29]



High Precision Low Recall

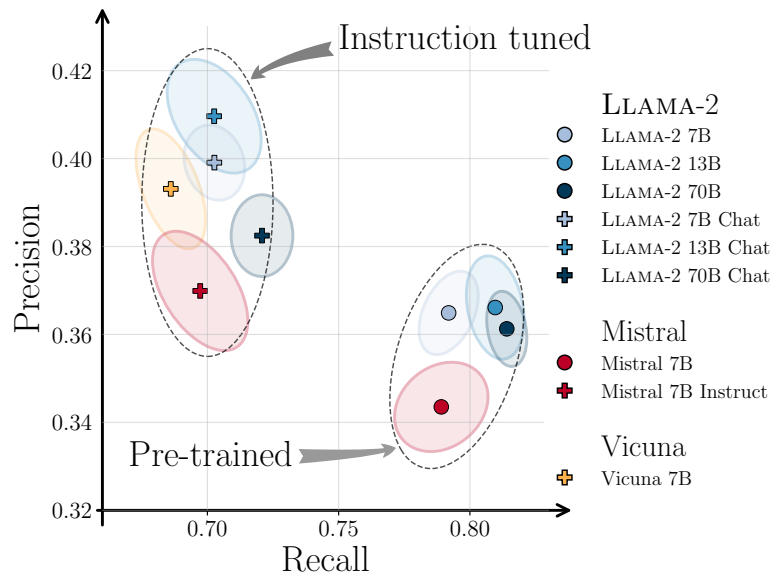


Precision: 0.80 Recall: 0.70

Precision and Recall for Generative Models

For LLMs

On open-ended generation, the quality and diversity of LLMs can also be evaluated using Precision and Recall: Bronnec et al. [8]

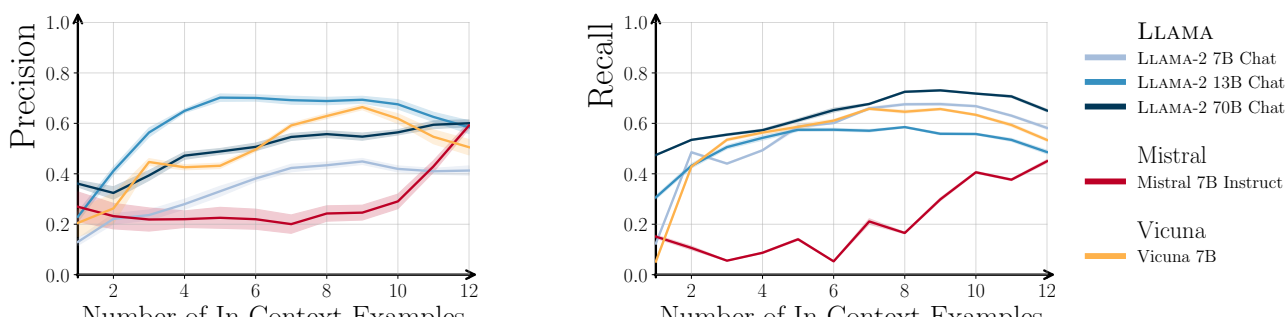


Le Bronnec, Vérine et al. · ACL 2024

Precision and Recall for Generative Models

For LLMs

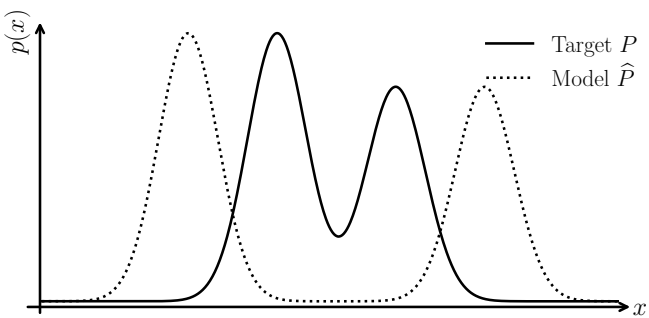
We can also evaluate the quality and diversity of LLMs on Chatbot open-ended generation. We can for instance check the impact of In-Context examples on the quality and diversity of the generated text. For instance on Wikipedia Biographies generation:

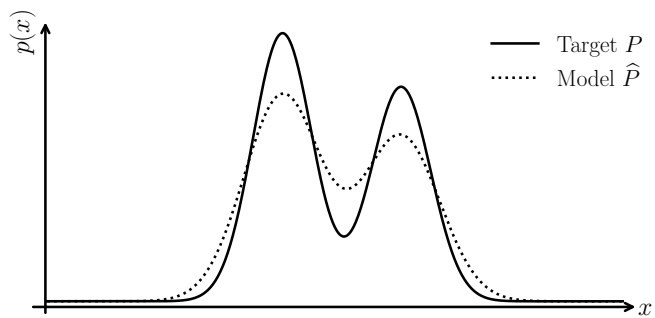


Le Bronnec, Vérine et al. · ACL 2024

Precision and Recall for Generative Models

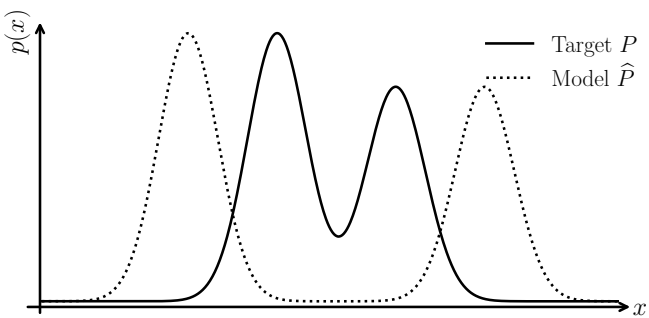
For infinite support

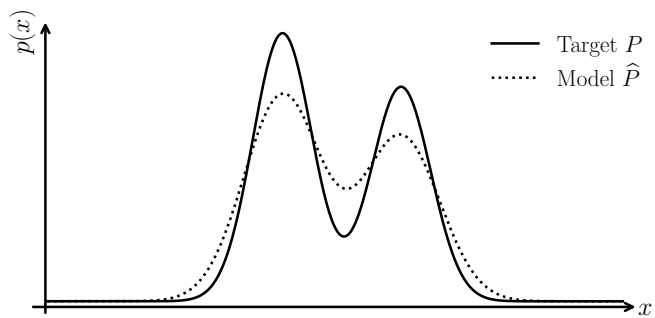




Precision and Recall for Generative Models

For infinite support





Both distributions have **perfect** Precision *and* Recall.

PR-Curve for Generative Models

Definition

Definition 1.4 (PR-Curve for Generative Models - Sajjadi et al. [22], Simon et al. [24])

Let $P, \widehat{P} \in \mathcal{P}(\mathcal{X})$ be two distributions such that $P, \widehat{P} \ll \mu$. The PR-Curve is the set $\mathrm{PRD}(P, \widehat{P})$ defined as:

$$\mathrm{PRD}(P, \widehat{P}) = \{(\alpha_\lambda, \beta_\lambda) \mid \lambda \in [0, \infty]\}$$

with:

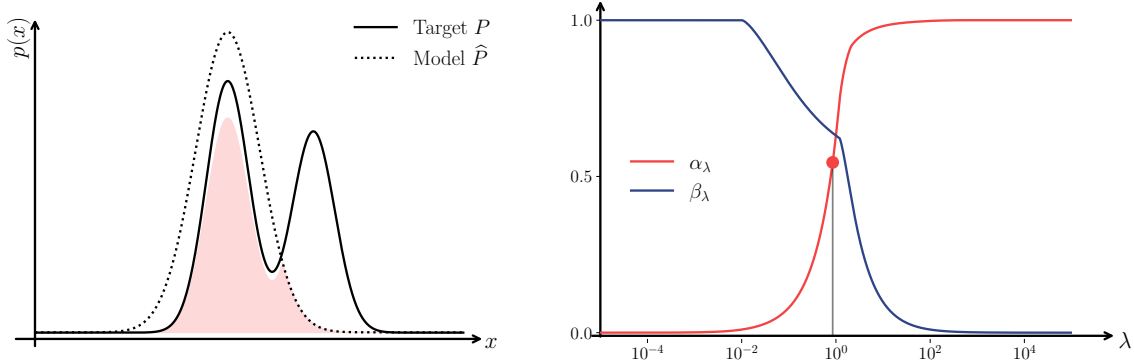
$$\alpha_\lambda = \int_{\mathcal{X}} \min\left(\lambda p(\boldsymbol{x}), \widehat{p}(\boldsymbol{x})\right) \mathrm{d}\mu(\boldsymbol{x}) \quad \textit{and} \quad \beta_\lambda = \int_{\mathcal{X}} \min\left(p(\boldsymbol{x}), \widehat{p}(\boldsymbol{x})/\lambda\right) \mathrm{d}\mu(\boldsymbol{x}).$$

PR-Curve for Generative Models

Definition

For the Precision, λp is compared to $\widehat{p}$ for different threshold $\lambda \in [0, +\infty]$:

$$\alpha_\lambda = \int_{\mathcal{X}} \min\left(\lambda p(\boldsymbol{x}), \widehat{p}(\boldsymbol{x})\right) \mathrm{d}\mu(\boldsymbol{x}) \tag{1}$$

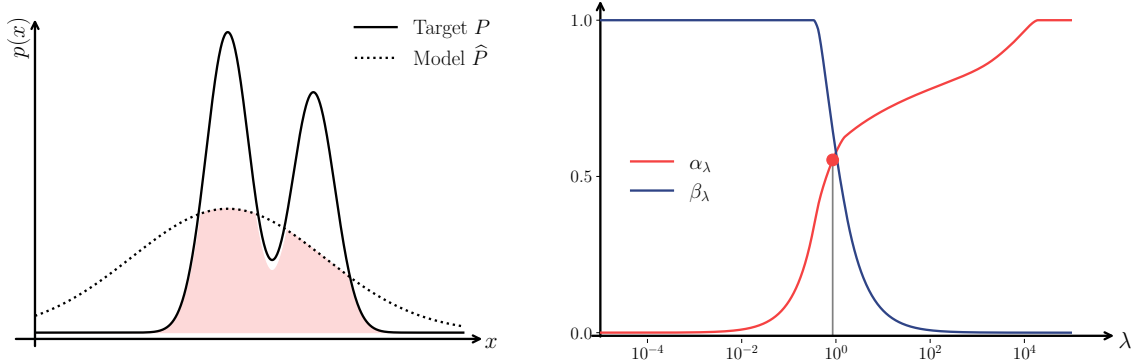


PR-Curve for Generative Models

Definition

For the Precision, λp is compared to $\widehat{p}$ for different threshold $\lambda \in [0, +\infty]$:

$$\alpha_\lambda = \int_{\mathcal{X}} \min\left(\lambda p(\boldsymbol{x}), \widehat{p}(\boldsymbol{x})\right) \mathrm{d}\mu(\boldsymbol{x}) \tag{1}$$

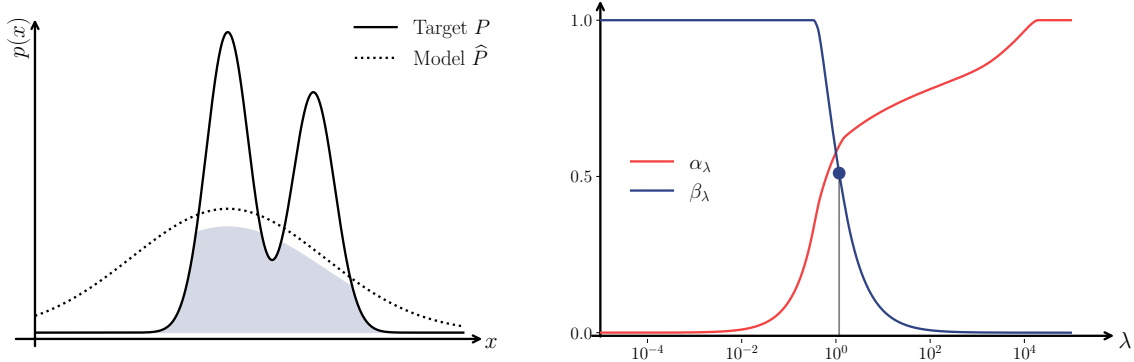


PR-Curve for Generative Models

Definition

For the Recall, p is compared to $\widehat{p}/\lambda$ for different threshold $\lambda \in [0, +\infty]$:

$$\beta_\lambda = \int_{\mathcal{X}} \min\left(p(\boldsymbol{x}), \widehat{p}(\boldsymbol{x})/\lambda\right) \mathrm{d}\mu(\boldsymbol{x}) \tag{2}$$

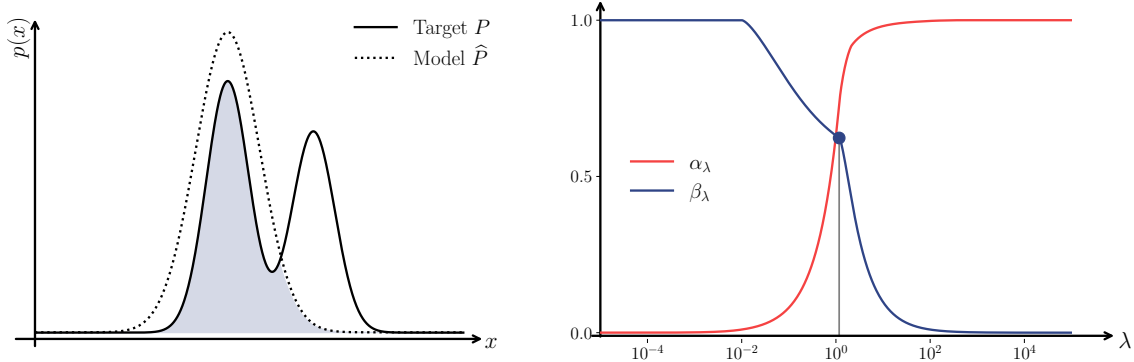


PR-Curve for Generative Models

Definition

For the Recall, p is compared to $\widehat{p}/\lambda$ for different threshold $\lambda \in [0, +\infty]$:

$$\beta_\lambda = \int_{\mathcal{X}} \min\left(p(\boldsymbol{x}), \widehat{p}(\boldsymbol{x})/\lambda\right) \mathrm{d}\mu(\boldsymbol{x}) \tag{2}$$



PR-Curve for Generative Models

Examples

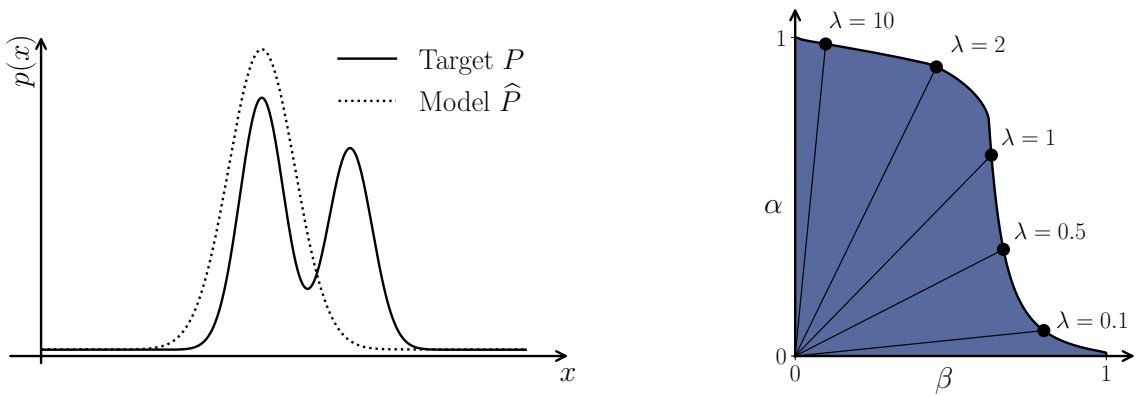


Figure. Learning distribution with low recall and high precision.

PR-Curve for Generative Models

Examples

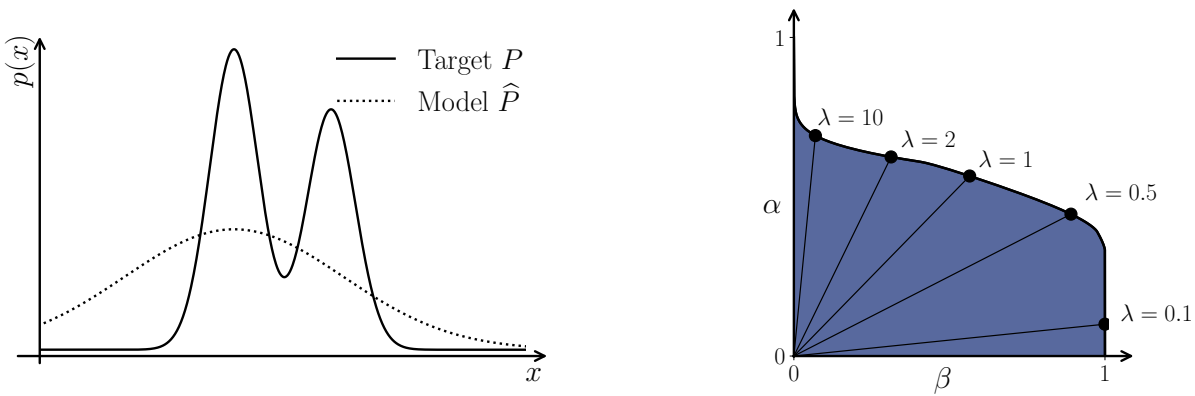


Figure. Learning distribution with high recall and low precision.

PR-Curve for Generative Models

Examples

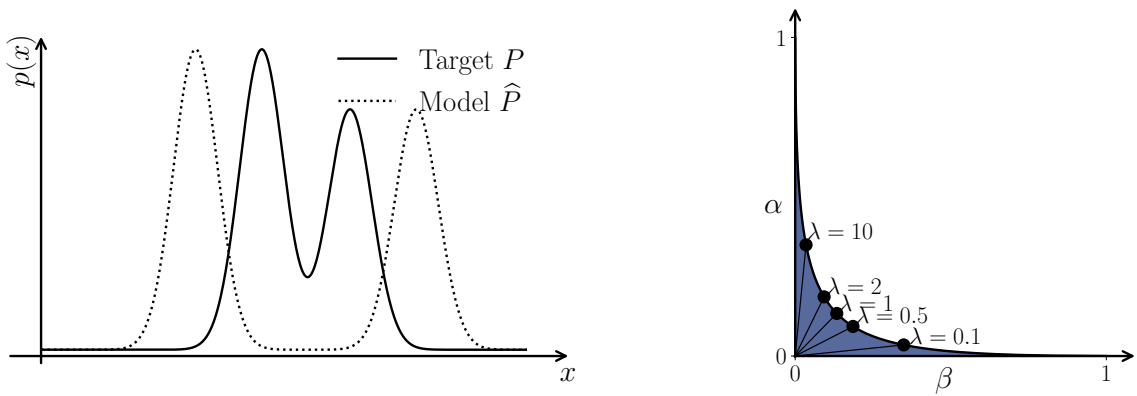


Figure. Learning distribution with low recall and low precision.

PR-Curve for Generative Models

Examples

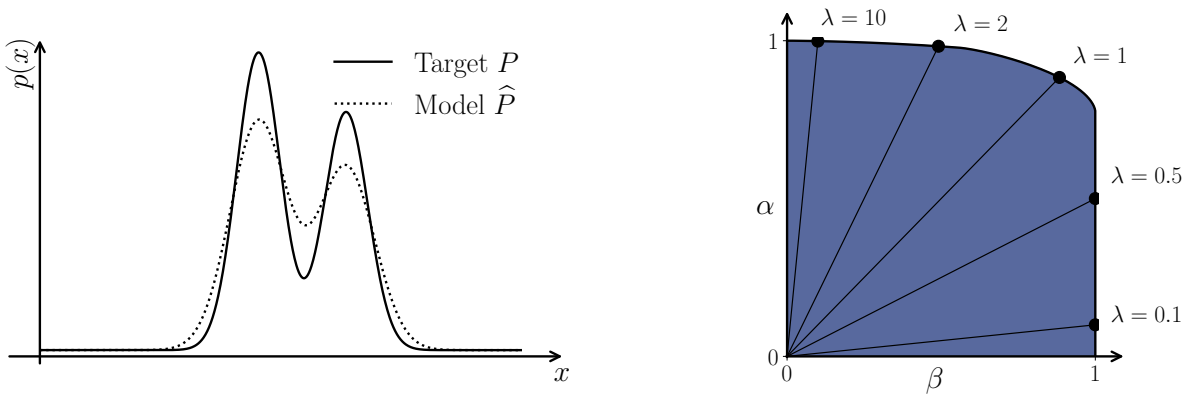


Figure. Learning distribution with high recall and high precision.

The PR-Curve is a generalization of the Precision and Recall for finite support:

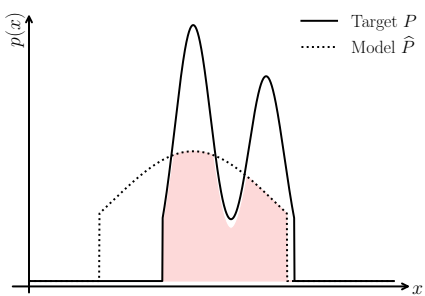
Theorem 1.5 (Support-based and PR-Curves - Siry et al. [25])

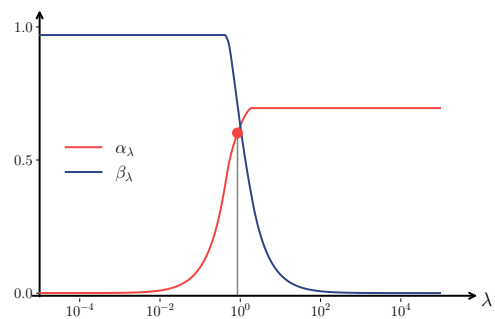
Let $P, \widehat{P} \in \mathcal{P}(\mathcal{X})$ be two distributions. Then, the support-based Precision and Recall $(\bar{\alpha}, \bar{\beta})$ are related to the PR-Curve values $\mathrm{PRD}(P, \widehat{P})$ *for $\lambda = 0$ and $\lambda = \infty$:*

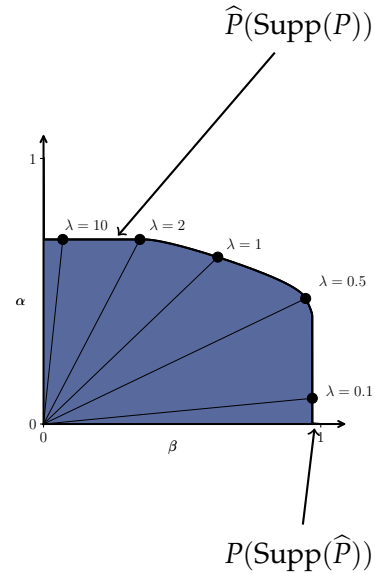
$$\bar{\alpha} = \max_{\lambda} \alpha_\lambda = \alpha_\infty \quad \textit{and} \quad \bar{\beta} = \max_{\lambda} \beta_\lambda = \beta_0.$$

PR-Curve and Support-Based Precision and Recall

Relation

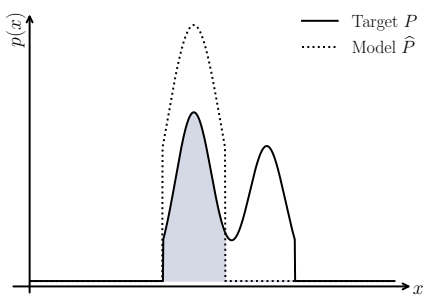


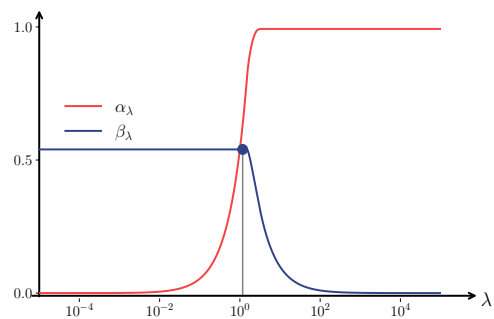


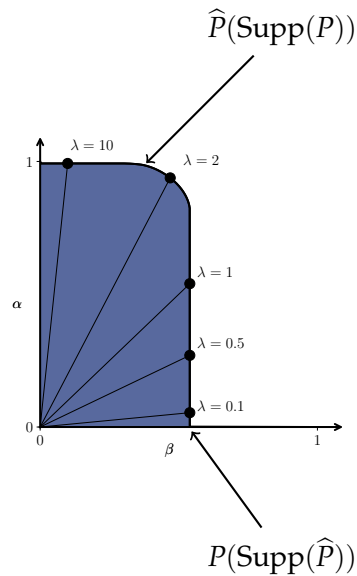


PR-Curve and Support-Based Precision and Recall

Relation



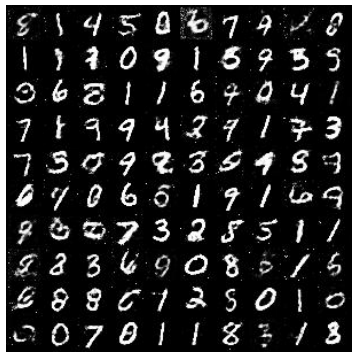




PR-Curve for Generative Models

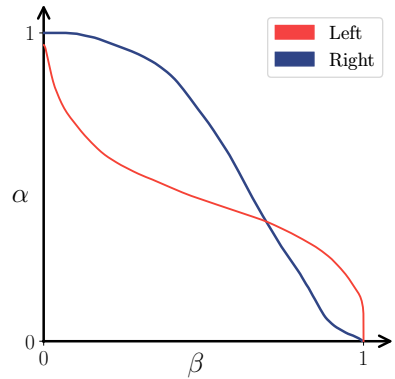
In Practice

Low Precision High Recall



Precision: 0.54 Recall: 0.91

PR-Curves



High Precision Low Recall



Precision: 0.80 Recall: 0.70

PR-Curve for Generative Models

In NLP

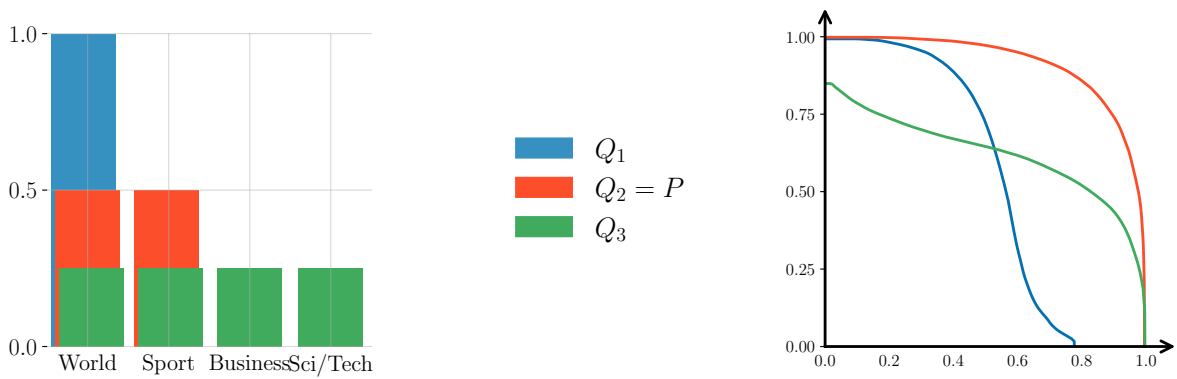


Figure. PR-Curve for distributions journal articles: AG News.

On the Platform

Metrics used to evaluate your models are:

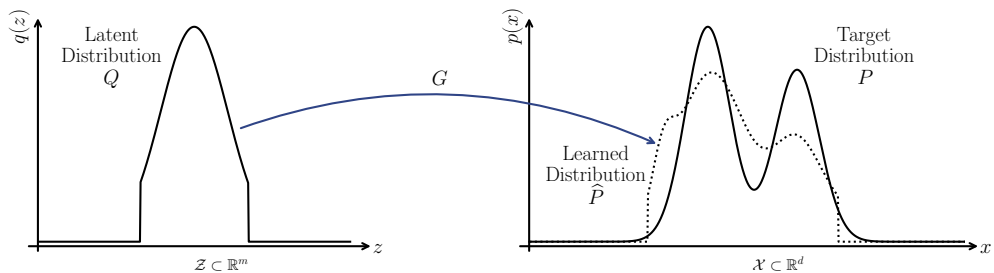
- FID
- Precision (for finite support)
- Recall (for finite support)
- (Obviously) the visual inspection of the generated samples.

Tuning:
How can we tune a model to a specific trade-off between Precision and Recall?

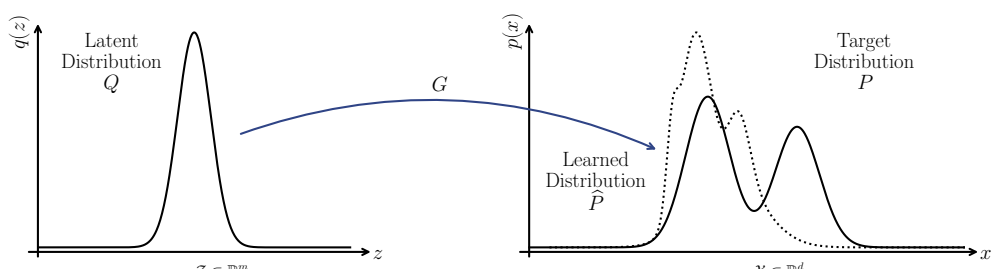
Tuning Precision and Recall in Generative Models

Truncation

Hard Truncation
Karras et al. [16]



Soft Truncation
Kingma and Dhariwal [17]



Tuning Precision and Recall in Generative Models

Hard Truncation



Figure. From left to right: $\psi = 0.0$, $\psi = 0.3$ $\psi = 0.7$ $\psi = 1.0$.

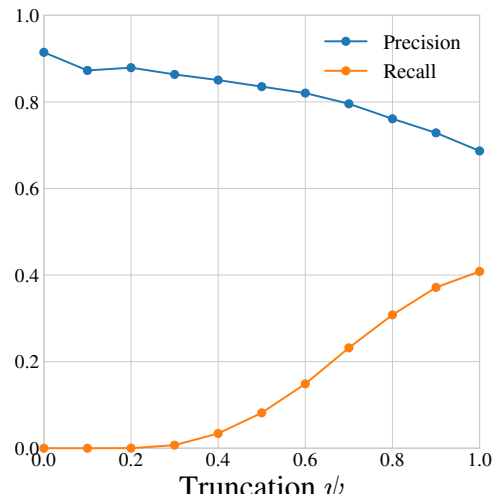


Figure. Source: [18]



(a) $\psi = 0.04$ (b) $\psi = 0.5$ (c) $\psi = 1.0$ (d) $\psi = 2.0$

Figure. Soft-Truncation on BigGAN. Source:[7].

Tuning Precision and Recall in Generative Models

Temperature scaling

For a token distribution $\widehat{P}(x_l|\boldsymbol{x}_{<l})$, temperature $t > 0$ rescales the probabilities:

$$\widehat{P}^t(x_l|\boldsymbol{x}_{<l}) = \frac{\widehat{P}(x_l|\boldsymbol{x}_{<l})^{1/t}}{\sum_{v\in\mathcal{V}} \widehat{P}(v|\boldsymbol{x}_{<l})^{1/t}}.$$

Tuning Precision and Recall in Generative Models

Temperature scaling

For a token distribution $\widehat{P}(x_l|\boldsymbol{x}_{<l})$, temperature $t > 0$ rescales the probabilities:

$$\widehat{P}^t(x_l|\boldsymbol{x}_{<l}) = \frac{\widehat{P}(x_l|\boldsymbol{x}_{<l})^{1/t}}{\sum_{v\in\mathcal{V}} \widehat{P}(v|\boldsymbol{x}_{<l})^{1/t}}.$$

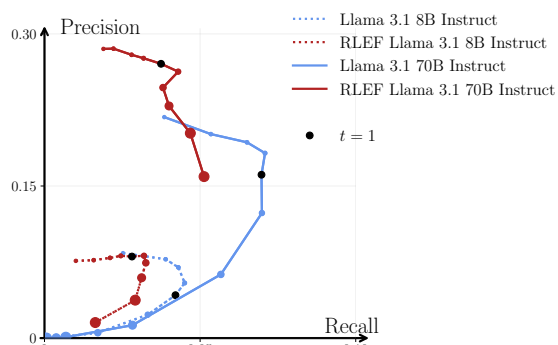
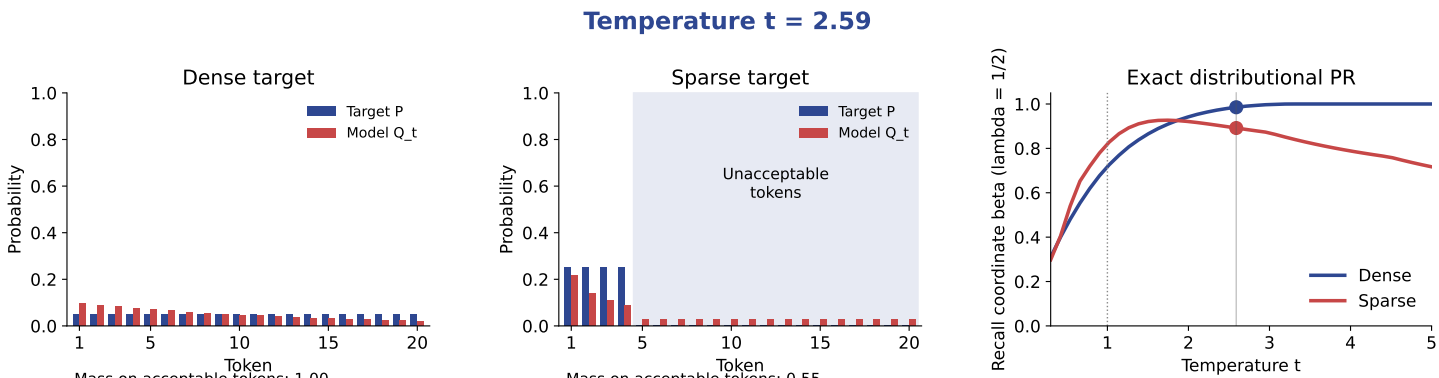


Figure. Effect on the PR-Curve for different temperatures for Llama 3.1 on CodeContests. Source: Verine et al. [28].

Temperature: dense versus sparse targets

A controlled illustration



$Q_t(x) \propto Q(x)^{1/t}$: increasing t flattens *all* probabilities, including those of unacceptable tokens.

Synthetic one-token example; exact PR coordinate $\beta_{1/2} = 2\sum_x \min\{P(x)/2, Q_t(x)\}$.

A theoretical limit of temperature scaling

Sparsity matters

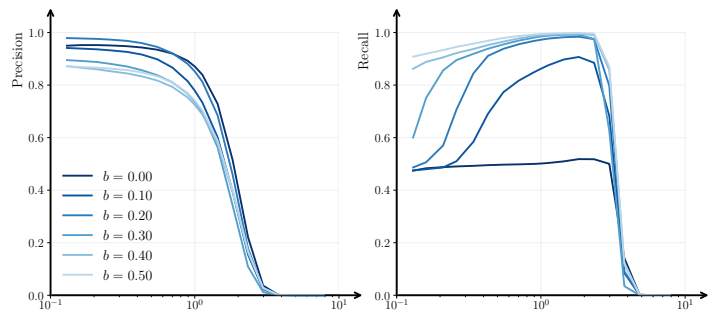
A finite sequence model

Vocabulary size V, sequence length L, target P and model Q_t. Let Z bound the model's within-context logit range.

For each fixed $\lambda > 0$,

$$\alpha_\lambda(P\|Q_t) \leq \frac{|\operatorname{supp}(P)|}{V^L} e^{ZL/t},$$

$$\beta_\lambda(P\|Q_t) \leq \frac{1}{\lambda} \frac{|\operatorname{supp}(P)|}{V^L} e^{ZL/t}.$$

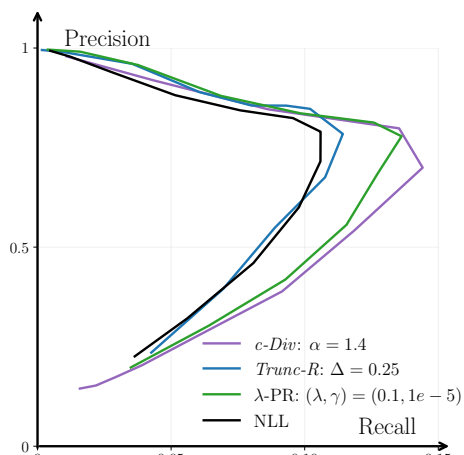


In the multiplication experiment, recall can first rise, then fall. At high temperature, probability leaks to unacceptable sequences.

A statement about distributional PR coordinates, not the nearest-neighbour image metric. High temperature is not a guarantee of coverage.

If decoding is not enough, change the loss

Training and inference are complementary



- NLL plus temperature is only one approach.
- c-Div and λ-PR change the learning objective.
- Train toward coverage; use decoding to control quality.

Here: controlled integer-multiplication experiments.

Next: how the choice of divergence changes generative training.

Training

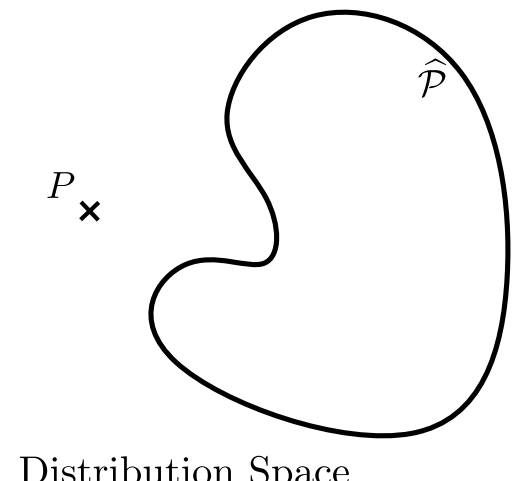
Changing the objective to shape precision and recall

Training a Generative Model

in general

Traditionally, the goal is to minimize *a dissimilarity measure* between the target distribution P and the learned distribution $\widehat{P}$:

$$\min_{G} \; D(P, \widehat{P}_G) \tag{3}$$

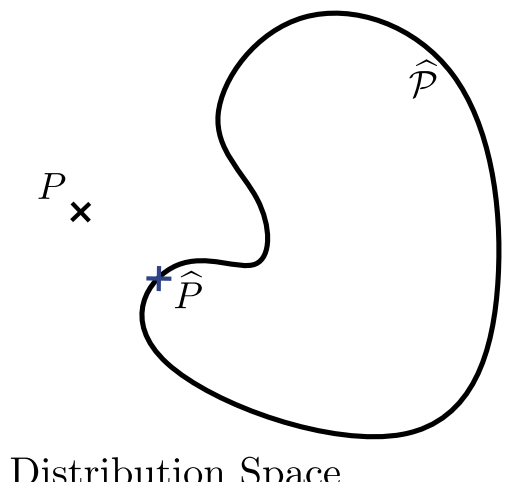


Training a Generative Model

with f-divergences

Traditionally, the goal is to minimize *an f-divergence* between the target distribution P and the learned distribution $\widehat{P}$:

$$\min_G \ \mathcal{D}_f(P \| \widehat{P}_G) \tag{3}$$



Definition 2.1 (f-divergences)

For any two probability distributions P and $\widehat{P}$ in $\mathcal{P}(\mathcal{X})$ such that $P, \widehat{P} \ll \mu$. Let p and $\widehat{p}$ be the Radon-Nikodym densities of P and $\widehat{P}$ with respect to μ, respectively. Let f be any convex lower semi-continuous function $f : [0, \infty] \to]-\infty, +\infty]$ such that $f(1) = 0$, the f-divergence between P and $\widehat{P}$ is

$$\mathcal{D}_f(P\|\widehat{P}) = \int_{\mathcal{X}} \widehat{p}(\boldsymbol{x}) f\left(\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}\right) \mathrm{d}\mu(\boldsymbol{x}). \tag{4}$$

Definition 2.1 (f-divergences)

For any two probability distributions P and $\widehat{P}$ in $\mathcal{P}(\mathcal{X})$ such that $P, \widehat{P} \ll \mu$. Let p and $\widehat{p}$ be the Radon-Nikodym densities of P and $\widehat{P}$ with respect to μ, respectively. Let f be any convex lower semi-continuous function $f : [0, \infty] \to]-\infty, +\infty]$ such that $f(1) = 0$, the f-divergence between P and $\widehat{P}$ is

$$\mathcal{D}_f(P\|\widehat{P}) = \int_{\mathcal{X}} \widehat{p}(\boldsymbol{x}) f\left(\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}\right) \mathrm{d}\mu(\boldsymbol{x}). \tag{4}$$

Usual divergences are f-divergences:

- Kullback-Leibler (KL),
- Reverse Kullback-Leibler (rKL),
- Jensen-Shannon (JS),
- Total Variation (TV),
- $\alpha-$divergences.

Estimating f-divergences

Dual variational form

f-divergences are *hardly tractable* in practice. However, they can be approximated by a dual approximation.

Estimating f-divergences

Dual variational form

f-divergences are *hardly tractable* in practice. However, they can be approximated by a dual approximation.

- $f^*(t) = \sup_{u \in \mathbb{R}} \{tu - f(u)\}$ be *the Fenchel conjugate* of f.
- $\mathcal{T}$ be the set of all measurable functions $\mathcal{X} \to \mathbb{R}$.

Estimating f-divergences

Dual variational form

f-divergences are *hardly tractable* in practice. However, they can be approximated by a dual approximation.

- $f^*(t) = \sup_{u \in \mathbb{R}} \{tu - f(u)\}$ be *the Fenchel conjugate* of f.
- $\mathcal{T}$ be the set of all measurable functions $\mathcal{X} \to \mathbb{R}$.

Theorem 2.2 (Dual variational form of an f-divergence- Nguyen et al. [19])

Let $P, \widehat{P} \in \mathcal{P}(\mathcal{X})$ two distributions such that P is absolutely continuous with respect to $\widehat{P}$ and f a suitable generator function. The f-divergence between P and $\widehat{P}$ admits a dual variational form:

$$\mathcal{D}_f(P \| \widehat{P}) = \sup_{T \in \mathcal{T}} \left(\mathbb{E}_{\boldsymbol{x} \sim P} [T(\boldsymbol{x})] - \mathbb{E}_{\boldsymbol{x} \sim \widehat{P}} [f^*(T(\boldsymbol{x}))] \right). \tag{5}$$

We use $T^{\text{opt}} \in \mathcal{T}$ to denote the function that achieves the supremum.

By doing so, we can rewrite the optimization problem as:

$$\min_G \max_T \underbrace{\mathbb{E}_{\boldsymbol{x}\sim P}\left[T(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{x}\sim \widehat{P}_G}\left[f^*(T(\boldsymbol{x}))\right]}_{\mathcal{D}_{f,T}^{\text{dual}}} \tag{6}$$

Training a Generative Model with f-divergences

Using the dual variational form

By doing so, we can rewrite the optimization problem as:

$$\min_G \max_T \underbrace{\mathbb{E}_{\boldsymbol{x}\sim P}\left[T(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{x}\sim \widehat{P}_G}\left[f^*(T(\boldsymbol{x}))\right]}_{\mathcal{D}_{f,T}^{\text{dual}}} \tag{6}$$

- The discriminator T is trained *to estimate* the divergence.
- The generator G is trained *to minimize* the divergence.

Training a Generative Model with f-divergences

Using the dual variational form

By doing so, we can rewrite the optimization problem as:

$$\min_G \max_T \underbrace{\mathbb{E}_{\boldsymbol{x}\sim P}\left[T(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{x}\sim \widehat{P}_G}\left[f^*(T(\boldsymbol{x}))\right]}_{\mathcal{D}_{f,T}^{\text{dual}}} \tag{6}$$

- The discriminator T is trained *to estimate* the divergence.
- The generator G is trained *to minimize* the divergence.

Training a Generative Model with f-divergences

Using the dual variational form

By doing so, we can rewrite the optimization problem as:

$$\min_G \max_T \; \mathbb{E}_{\boldsymbol{x}\sim P}\left[\log\left(D(\boldsymbol{x})\right)\right] - \mathbb{E}_{\boldsymbol{x}\sim \widehat{P}_G}\left[f^*\left(\log(D(\boldsymbol{x}))\right)\right] \tag{6}$$

- The discriminator T is trained *to estimate* the divergence.
- The generator G is trained *to minimize* the divergence.
- With $T(\boldsymbol{x}) = \log(D(\boldsymbol{x}))$ with $D(\boldsymbol{x}) \in [0, 1]$.

Training a Generative Model with f-divergences

Using the dual variational form

By doing so, we can rewrite the optimization problem as:

$$\min_G \max_T \mathbb{E}_{\boldsymbol{x}\sim P}\left[\log\left(D(\boldsymbol{x})\right)\right] + \mathbb{E}_{\boldsymbol{x}\sim \widehat{P}_G}\left[\log\left(1 - D(\boldsymbol{x})\right)\right] \tag{6}$$

- The discriminator T is trained *to estimate* the divergence.
- The generator G is trained *to minimize* the divergence.
- With $T(\boldsymbol{x}) = \log(D(\boldsymbol{x}))$ with $D(\boldsymbol{x}) \in [0, 1]$.
- $f^*(t) = f^*_{\text{JS}}(t) = -\log(1 - \exp(t))$ for the Jensen-Shannon divergence.

We recover the original GAN framework.

Training a Generative Model with f-divergences

Using the dual variational form

By doing so, we can rewrite the optimization problem as:

$$\min_G \max_T \underbrace{\mathbb{E}_{\boldsymbol{x}\sim P}\left[T(\boldsymbol{x})\right] - \mathbb{E}_{\boldsymbol{x}\sim \widehat{P}_G}\left[f^*(T(\boldsymbol{x}))\right]}_{\mathcal{D}^{\text{dual}}_{f,T}} \tag{6}$$

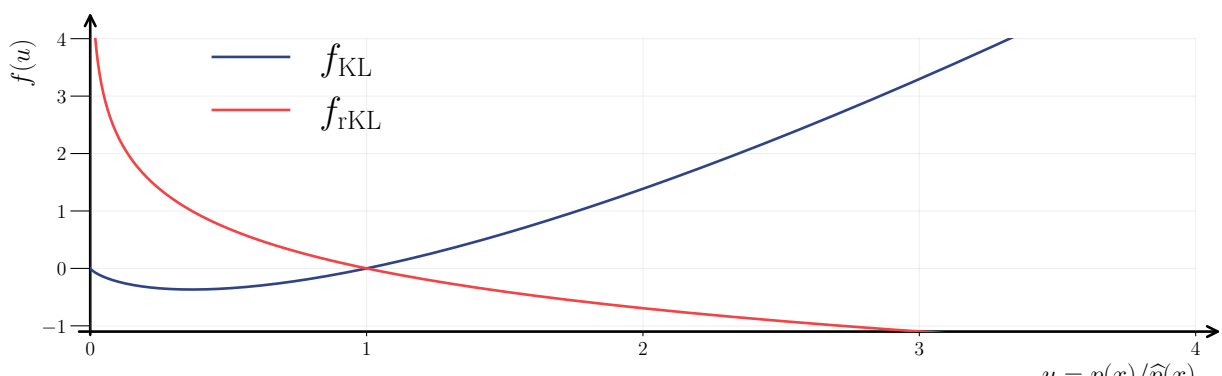
- The discriminator T is trained *to estimate* the divergence.
- The generator G is trained *to minimize* the divergence.

- Generative Adversarial Networks [12] for *the Jensen-Shannon divergence*.
- Extended to other f-divergences by Nowozin et al. [20].
- Extend to other generative models such as Normalizing Flows by Grover et al. [13].

Effect of the f-divergence on the learned distribution

All f-divergences are not equal:

$$\mathcal{D}_f(P\|\widehat{P}) = \mathbb{E}_{\boldsymbol{x}\sim\widehat{P}}\left[f\left(\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}\right)\right]$$

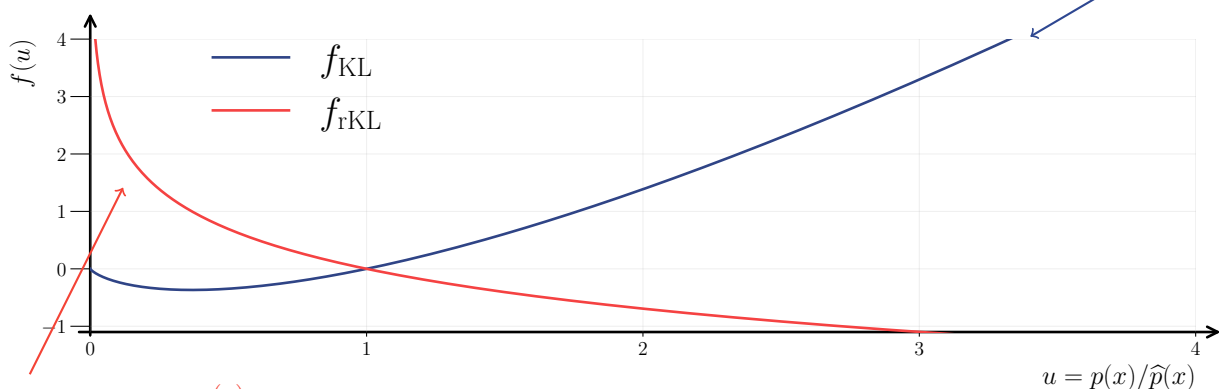


Effect of the f-divergence on the learned distribution

All f-divergences are not equal:

$$\mathcal{D}_f(P\|\widehat{P}) = \mathbb{E}_{\boldsymbol{x}\sim\widehat{P}}\left[f\left(\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}\right)\right]$$

Penalizing high values of $\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}$



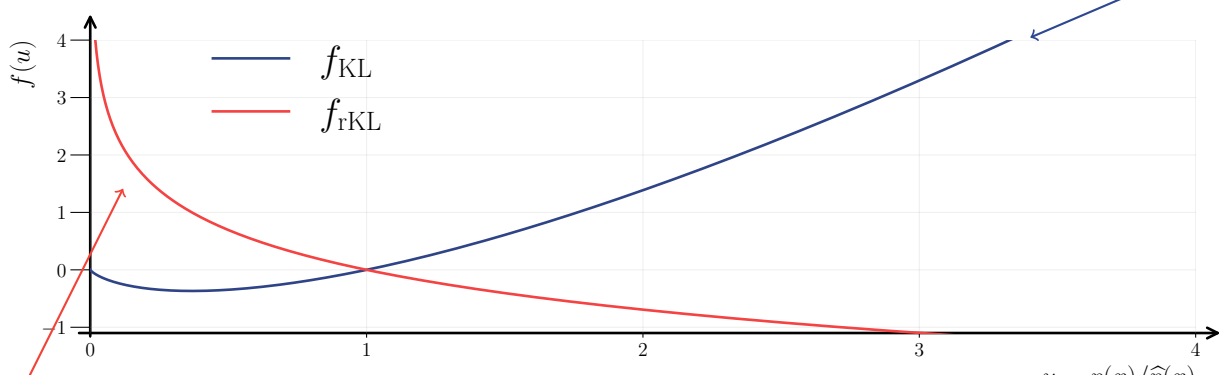
Penalizing low values of $\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}$

Effect of the f-divergence on the learned distribution

All f-divergences are not equal:

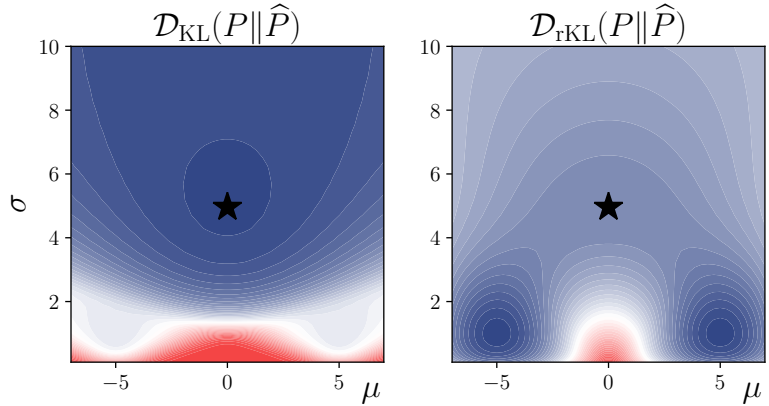
$$\mathcal{D}_f(P\|\widehat{P}) = \mathbb{E}_{\boldsymbol{x}\sim\widehat{P}}\left[f\left(\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}\right)\right]$$

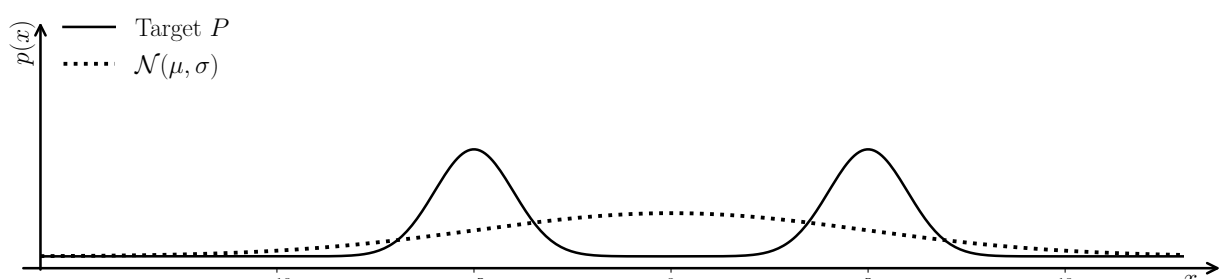
Penalizing high values of $\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}$
Favors high recall



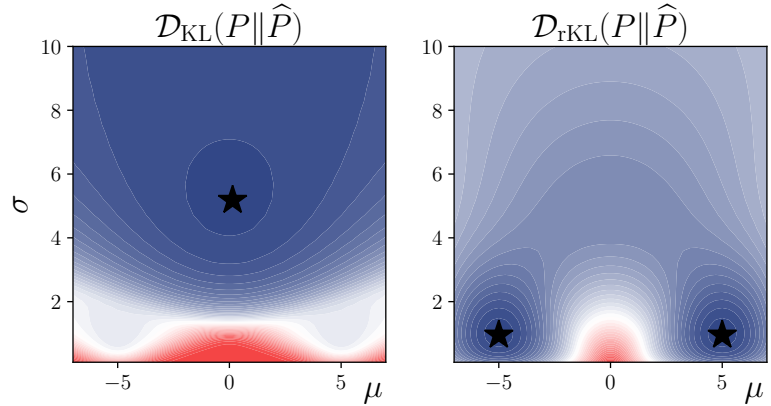
Penalizing low values of $\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}$
Favors high precision

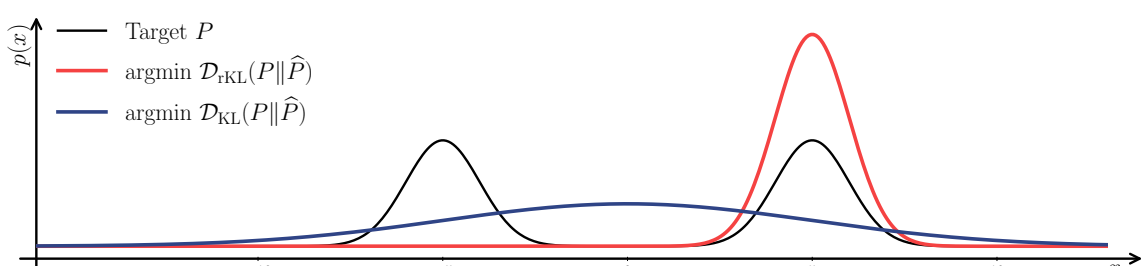
Examples of f-divergence minimization





Examples of f-divergence minimization





Can we optimize a specific trade-off between Precision and Recall?

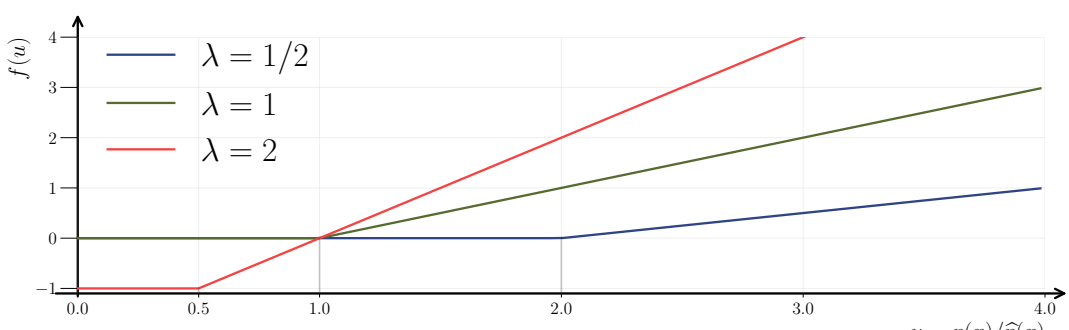
Can we optimize a specific trade-off between Precision and Recall?

- What is the relation between the Precision-Recall curve and f-divergences?

Definition 2.3 (PR-Divergence generator function f_λ)

Given a trade-off parameter $\lambda \in [0, +\infty]$, we define the generator function $f_\lambda : [0, +\infty] \to]-\infty, +\infty]$ given by

$$f_\lambda(u) = \begin{cases} \max(\lambda u, 1) - \max(\lambda, 1) & \text{for } \lambda \in [0, +\infty[, \\ \mathbb{1}_{\{u=0\}} & \text{for } \lambda = +\infty. \end{cases} \tag{7}$$



Proposition 2.4 (PR-Divergence)

For any distributions $P, \widehat{P} \in \mathcal{P}(\mathcal{X})$ such that $P, \widehat{P} \ll \mu$, then for any $\lambda \in [0, +\infty]$ the PR-Divergence defined as

$$\mathcal{D}_{\lambda\text{-PR}}(P \| \widehat{P}) = \int_{\mathcal{X}} \widehat{p}(\boldsymbol{x}) f_\lambda \left(\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})} \right) \mathrm{d}\mu(\boldsymbol{x}) \tag{8}$$

belongs to the class of f-divergences.

Precision-Recall Divergence

Linking the PR-Divergence to the PR-Curve

Theorem 2.5 (PR-Curves as a function of $\mathcal{D}_{\lambda\text{-PR}}$)

Given $P, \widehat{P} \in \mathcal{P}(\mathcal{X})$ such that $P, \widehat{P} \ll \mu$ and $\lambda \in [0, +\infty]$, the PR-Curve ∂PRD is related to the PR-Divergence $\mathcal{D}_{\lambda\text{-PR}}(P \| \widehat{P})$ as follows.

$$\alpha_\lambda(P \| \widehat{P}) = \min(1, \lambda) - \mathcal{D}_{\lambda\text{-PR}}(P \| \widehat{P}).$$
$$\beta_\lambda(P \| \widehat{P}) = \min(1, \lambda) - \mathcal{D}_{\lambda\text{-PR}}(\widehat{P} \| P).$$

Theorem 2.5 (PR-Curves as a function of $\mathcal{D}_{\lambda\text{-PR}}$)

Given $P, \widehat{P} \in \mathcal{P}(\mathcal{X})$ such that $P, \widehat{P} \ll \mu$ and $\lambda \in [0, +\infty]$, the PR-Curve ∂PRD is related to the PR-Divergence $\mathcal{D}_{\lambda\text{-PR}}(P\|\widehat{P})$ as follows.

$$\alpha_\lambda(P\|\widehat{P}) = \min(1, \lambda) - \mathcal{D}_{\lambda\text{-PR}}(P\|\widehat{P}).$$
$$\beta_\lambda(P\|\widehat{P}) = \min(1, \lambda) - \mathcal{D}_{\lambda\text{-PR}}(\widehat{P}\|P).$$

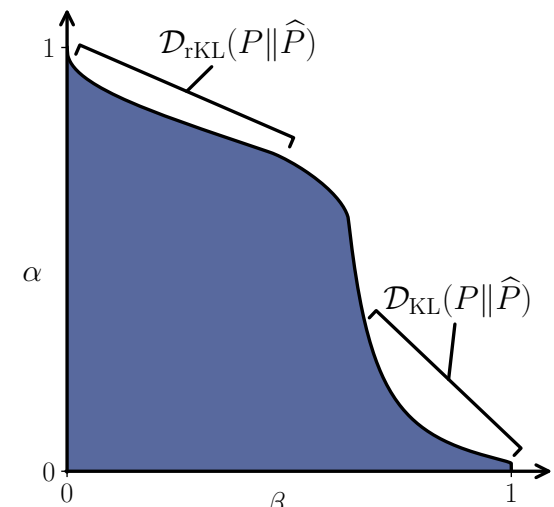
A direct consequence of Theorem 2.5:

$$\operatorname*{argmin}_{\widehat{P}\in\mathcal{P}(\mathcal{X})} \mathcal{D}_{\lambda\text{-PR}}(P\|\widehat{P}) = \operatorname*{argmax}_{\widehat{P}\in\mathcal{P}(\mathcal{X})} \alpha_\lambda(P\|\widehat{P}).$$

Theorem 2.6 (f-divergences as a weighted average of PR-Divergences)

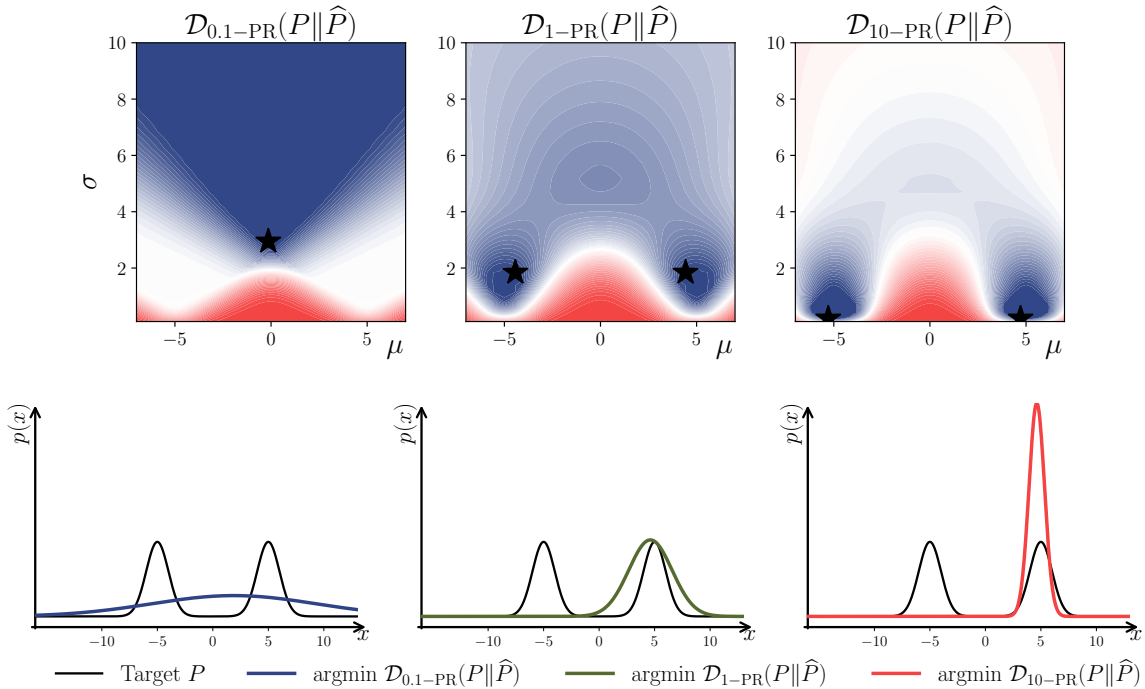
For any $P, \widehat{P} \in \mathcal{P}(\mathcal{X})$ supported on all $\mathcal{X}$ and any $\lambda \in [0, +\infty]$, then:

$$\mathcal{D}_f(P\|\widehat{P}) = \int_0^\infty \frac{1}{\lambda^3} f''\left(\frac{1}{\lambda}\right) \mathcal{D}_{\lambda\text{-PR}}(P\|\widehat{P}) \mathrm{d}\lambda,$$



Optimizing the PR-Divergence

Examples



Optimizing the PR-Divergence

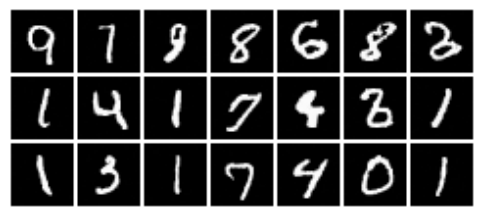
Examples

Optimizing the PR-Divergence with our approach

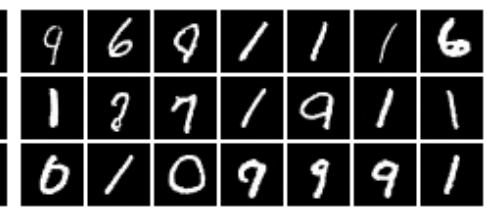
In Practice



(a) $\lambda = 0.1$



(b) $\lambda = 1$



(c) $\lambda = 10$



(d) $\lambda = 0.1$



(e) $\lambda = 1$



(f) $\lambda = 10$

Vérine et al. · NeurIPS 2023

Optimizing the PR-Divergence with our approach

Training GANs

Model	CIFAR-10 32×32			CelebA 64×64		
	FID	P	R	FID	P	R
Baseline Big-GAN	13.37	86.51	65.66	9.16	78.41	**51.42**
$\lambda = 0.05$	13.29	81.10	70.63	-	-	-
$\lambda = 0.1$	**11.62**	81.78	**74.58**	-	-	-
$\lambda = 0.2$	13.36	84.85	65.13	8.79	83.37	44.07
$\lambda = 0.5$	14.50	83.27	68.23	**6.03**	77.60	**55.98**
$\lambda = 1.0$	14.03	83.04	69.35	13.07	81.70	36.85
$\lambda = 2.0$	16.94	84.93	59.79	14.23	82.98	32.87
$\lambda = 5.0$	32.54	83.39	56.94	22.45	**83.96**	25.81
$\lambda = 10.0$	39.69	84.11	39.29	-	-	-
$\lambda = 20.0$	67.03	**90.03**	21.81	-	-	-

When λ increases, $\begin{cases} \text{Precision} \uparrow \\ \text{Recall} \downarrow \end{cases}$



$\lambda = 0.1$ $\lambda = 10$

Vérine et al. · NeurIPS 2023

Optimizing the PR-Divergence with our approach

Fine-Tuning GANs

Model	ImageNet 128 × 128			FFHQ 256 × 256		
	FID	P	R	FID	P	R
Baseline BigGAN	9.83	28.04	41.21	41.41	65.57	**10.17**
Soft $\psi = 0.7$	11.39	23.04	31.13	56.43	76.59	4.87
Soft $\psi = 0.5$	15.49	20.20	19.83	82.05	84.48	1.58
Hard $\psi = 2.0$	**9.69**	25.83	39.89	43.32	68.84	8.66
Hard $\psi = 1.0$	12.12	21.86	35.42	56.19	76.44	4.76
Hard $\psi = 0.5$	15.21	21.13	29.55	71.32	80.99	4.84
$\lambda = 0.2$	9.92	26.69	42.04	35.66	78.70	9.45
$\lambda = 0.5$	10.82	26.83	**42.38**	**35.24**	78.41	9.66
$\lambda = 1.0$	20.42	29.72	28.21	35.91	78.95	8.32
$\lambda = 2.0$	20.21	30.27	30.49	36.33	81.10	8.69
$\lambda = 5.0$	20.76	**30.87**	28.38	38.16	**84.31**	8.52

When λ increases, $\begin{cases} \text{Precision} \uparrow \\ \text{Recall} \downarrow \end{cases}$ with better performances than truncation.

Vérine et al. · NeurIPS 2023

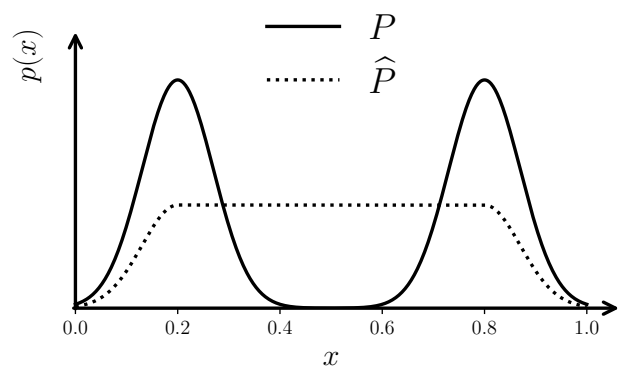
Improving:
How can we improve the quality and diversity of a pre-trained generative model?

Sampling from a generative model

General Setting

To sample a point from the learned distribution $\widehat{P}$:

- Sample $z \sim Q$.
- Compute $x = G(z)$.



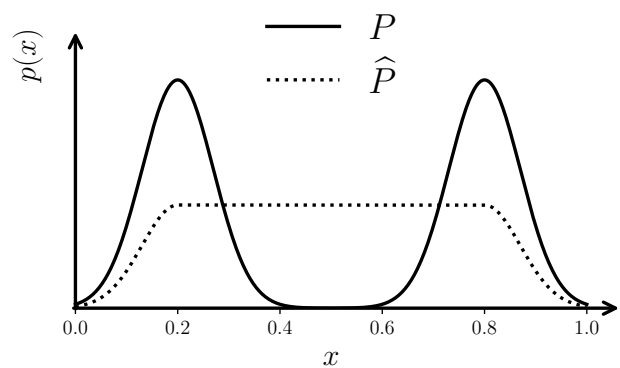
Sampling from a generative model

General Setting

To sample a point from the learned distribution $\widehat{P}$:

- Sample $\boldsymbol{z} \sim Q$.
- Compute $\boldsymbol{x} = G(\boldsymbol{z})$.

$$P \neq \widehat{P}$$



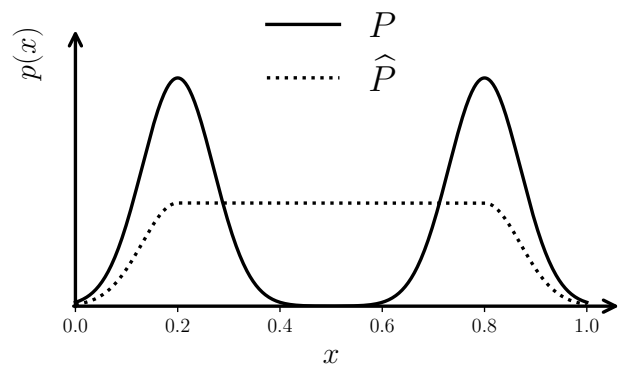
Sampling from a generative model

General Setting

To sample a point from the learned distribution $\widehat{P}$:

- Sample $\boldsymbol{z} \sim Q$.
- Compute $\boldsymbol{x} = G(\boldsymbol{z})$.

We have an estimation of $\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}$ using $\nabla f^*(T(\boldsymbol{x}))$.



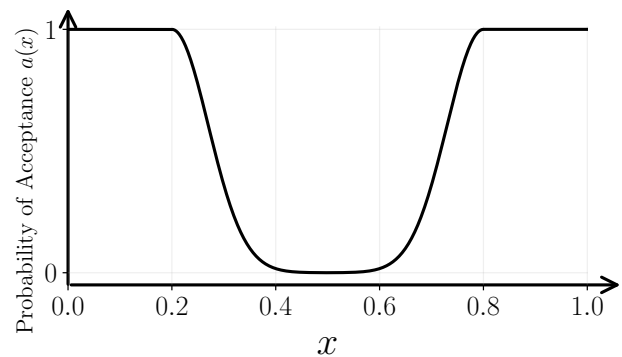
Sampling from a generative model

Rejection Sampling

To sample a point from the refined distribution $\widetilde{P}$:

- Sample $\boldsymbol{z} \sim Q$.
- Compute $\boldsymbol{x} = G(\boldsymbol{z})$.
- Accept $\boldsymbol{x}$ with probability $a(\boldsymbol{x})$.

Using $\frac{p(\boldsymbol{x})}{\widetilde{p}(\boldsymbol{x})}$ in $a(\boldsymbol{x})$ allows sampling from P.



Sampling from a generative model

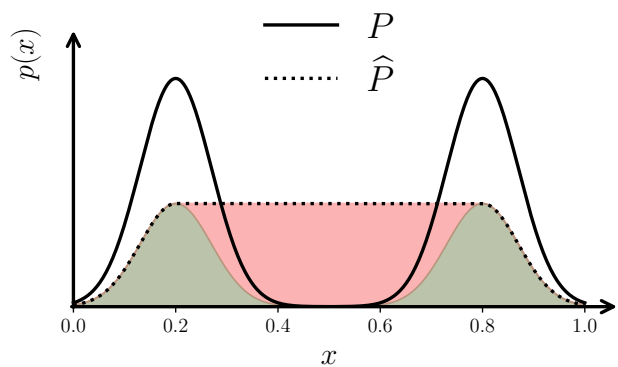
Rejection Sampling

To sample a point from the refined distribution $\widetilde{P}$:

- Sample $\boldsymbol{z} \sim Q$.
- Compute $\boldsymbol{x} = G(\boldsymbol{z})$.
- Accept $\boldsymbol{x}$ with probability $a(\boldsymbol{x})$.

The acceptance rate is :

$$\mathbb{E}_{\widehat{P}}\left[a(\boldsymbol{x})\right].$$



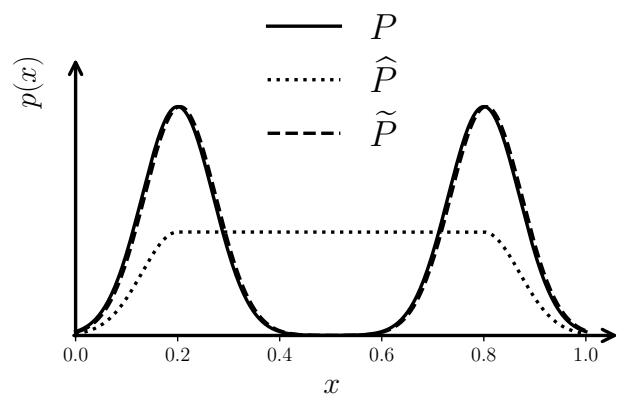
Sampling from a generative model

Rejection Sampling

To sample a point from the refined distribution $\widetilde{P}$:

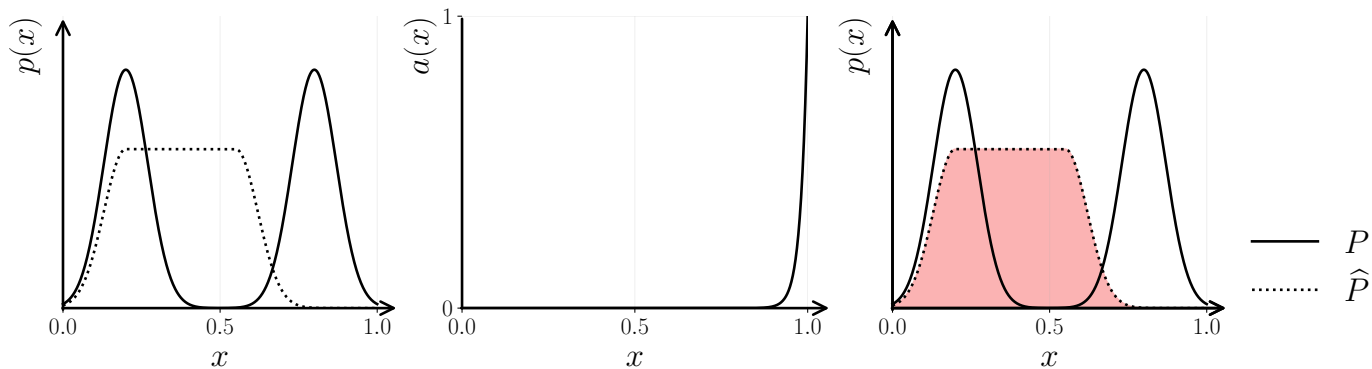
- Sample $\boldsymbol{z} \sim Q$.
- Compute $\boldsymbol{x} = G(\boldsymbol{z})$.
- Accept $\boldsymbol{x}$ with probability $a(\boldsymbol{x})$.

It defines a new distribution $\widetilde{P}$.



Sampling from a generative model

Rejection Sampling in high dimension



Acceptance rate: $4.20.10^{-6}$ %

Definition 3.1 (Discriminator Rejection Sampling (DRS) - Azadi et al. [2])

Let $\gamma \in \mathbb{R}$, the acceptance probability is:

$$a_{\text{DRS}}(\boldsymbol{x}) = \frac{r(\boldsymbol{x})}{r(\boldsymbol{x})\,(1 - e^{\gamma}) + Me^{\gamma}}.$$

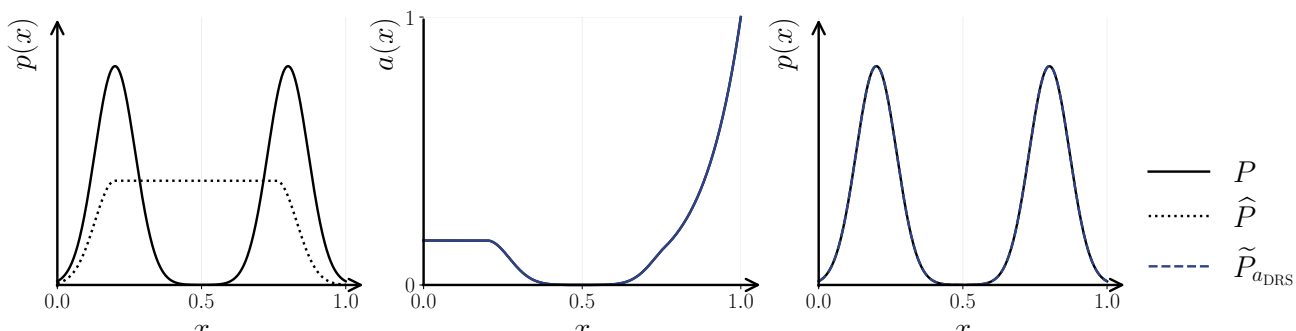
If $\gamma < 0$, then the acceptance rate increases.

Definition 3.1 (Discriminator Rejection Sampling (DRS) - Azadi et al. [2])

Let $\gamma \in \mathbb{R}$, the acceptance probability is:

$$a_{\text{DRS}}(\boldsymbol{x}) = \frac{r(\boldsymbol{x})}{r(\boldsymbol{x})\,(1 - e^{\gamma}) + Me^{\gamma}}.$$

If $\gamma < 0$, then the acceptance rate increases.



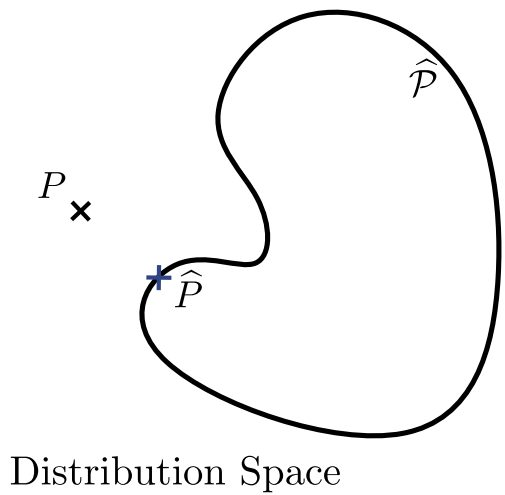
Acceptance rate: 7.91%

Optimal Budgeted Rejection Sampling

Definition

Traditionally, the goal is:

$$\min_{G} \quad \mathcal{D}_f(P \| \widehat{P}_G)$$



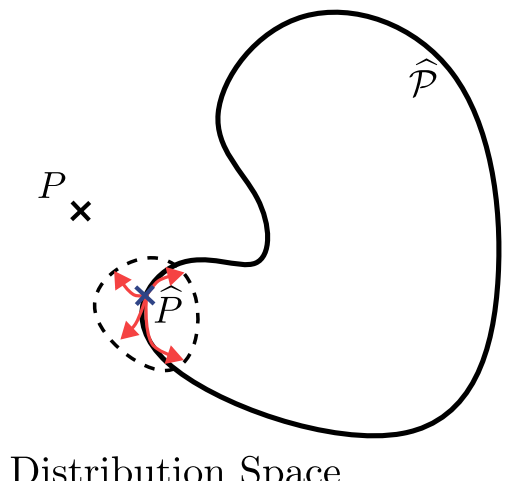
Optimal Budgeted Rejection Sampling

Definition

Traditionally, the goal is:

$$\min_{G} \quad \mathcal{D}_f(P \| \widehat{P}_G)$$

An acceptance function $a(\boldsymbol{x})$ such that the acceptance rate is greater than $1/K$ defines a refined distribution $\widetilde{P}_a$ in a convex set that contains $\widehat{P}_G$.

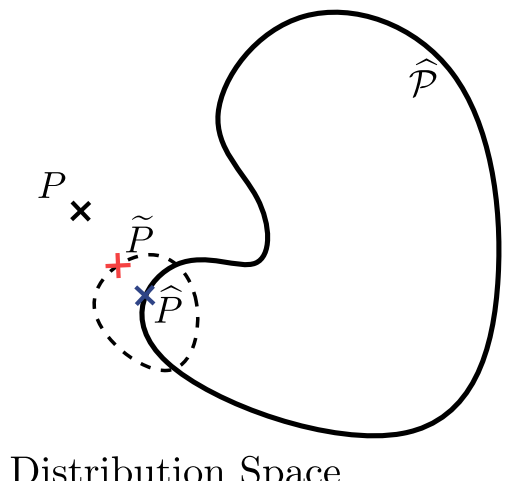


Optimal Budgeted Rejection Sampling

Definition

With a given $\widehat{P}_G$, our goal is:

$$\min_{a} \quad \mathcal{D}_f(P \| \widetilde{P}_a) \qquad \text{s.t.} \begin{cases} \mathbb{E}_{\widehat{P}}\left[a(\boldsymbol{x})\right] \geq 1/K, \\ \forall \boldsymbol{x} \in \mathcal{X},\ 0 \leq a(\boldsymbol{x}) \leq 1. \end{cases} \tag{9}$$



Optimal Budgeted Rejection Sampling

Definition

Theorem 3.2 (Optimal Acceptance Function)

For a sampling budget $K \geq 1$ and finite $\mathcal{X}$, the solution is,

$$a_{\text{OBRS}}(\boldsymbol{x}) = \min\left(\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}\frac{c_K}{M}, 1\right), \tag{10}$$

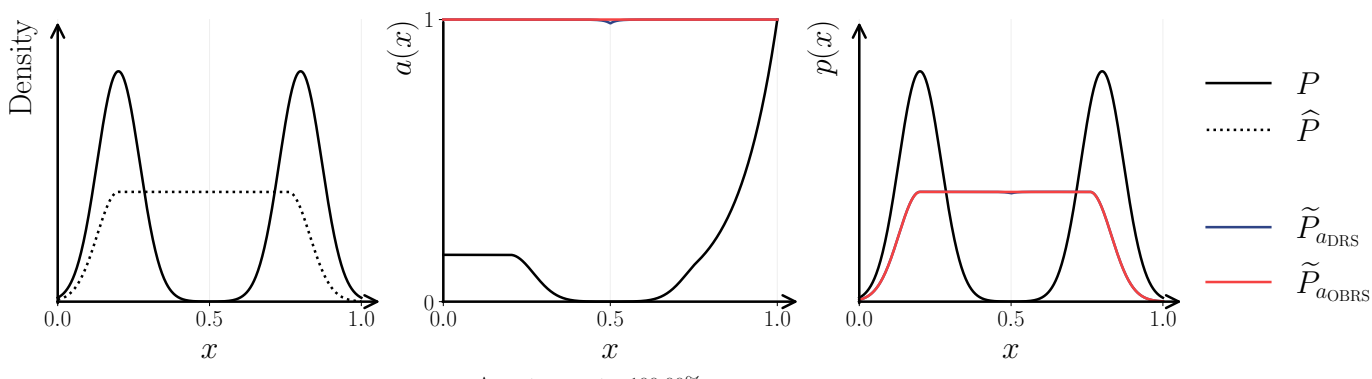
where $c_K \geq 1$ is such that $\mathbb{E}_{\boldsymbol{x}\sim\widehat{p}}[a_{\text{OBRS}}(\boldsymbol{x})] = 1/K$.

Theorem 3.2 (Optimal Acceptance Function)

For a sampling budget $K \geq 1$ and finite $\mathcal{X}$, the solution is,

$$a_{\text{OBRS}}(\boldsymbol{x}) = \min\left(\frac{p(\boldsymbol{x})}{\widehat{p}(\boldsymbol{x})}\frac{c_K}{M}, 1\right), \tag{10}$$

where $c_K \geq 1$ is such that $\mathbb{E}_{\boldsymbol{x}\sim\widehat{p}}[a_{\text{OBRS}}(\boldsymbol{x})] = 1/K$.



Improving Precision and Recall

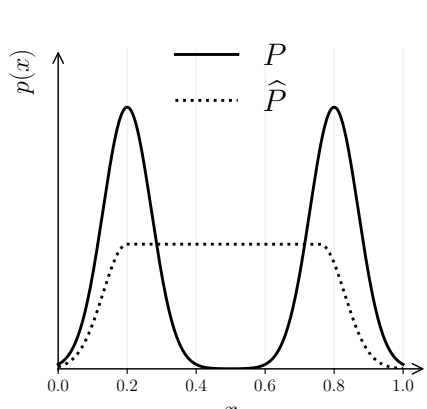
Effect of the OBRS

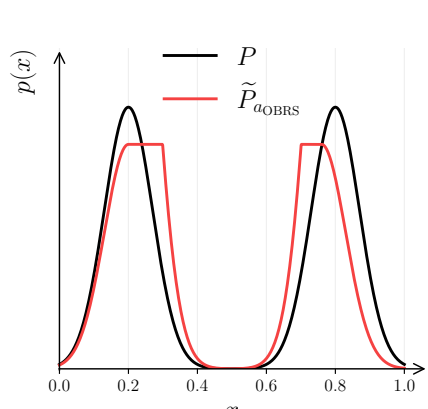
Proposition 3.3 (Precision and Recall Improvement)

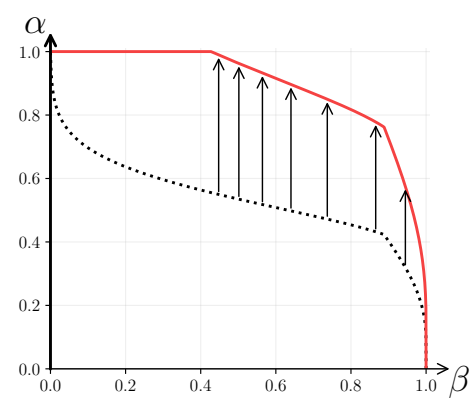
Let $K \leq M$ be the budget for the OBRS. For any $(\alpha, \beta) \in \mathrm{PRD}(P, \widehat{P})$ we have $(\alpha', \beta) \in \mathrm{PRD}(P, \widetilde{P}_{a_{\mathrm{OBRS}}})$ with $\alpha' = \min\{1, K\alpha\}$.

Proposition 3.3 (Precision and Recall Improvement)

Let $K \leq M$ be the budget for the OBRS. For any $(\alpha, \beta) \in \mathrm{PRD}(P, \widehat{P})$ we have $(\alpha', \beta) \in \mathrm{PRD}(P, \widetilde{P}_{a_{\mathrm{OBRS}}})$ with $\alpha' = \min\{1, K\alpha\}$.



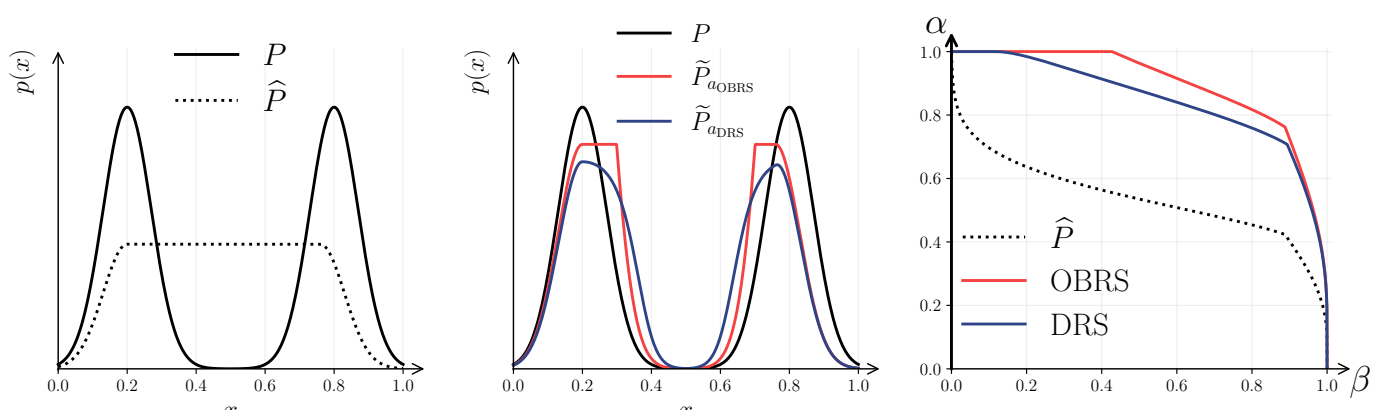




Acceptance rate: 55.54%

Proposition 3.3 (Precision and Recall Improvement)

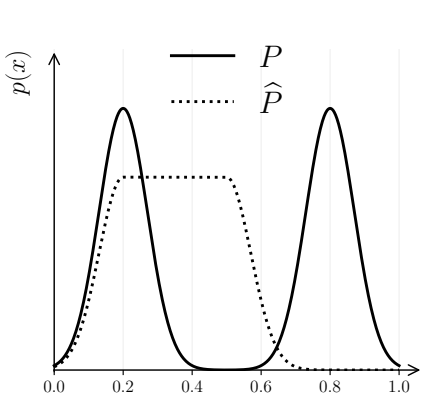
Let $K \leq M$ be the budget for the OBRS. For any $(\alpha, \beta) \in \mathrm{PRD}(P, \widehat{P})$ we have $(\alpha', \beta) \in \mathrm{PRD}(P, \widetilde{P}_{a_{\mathrm{OBRS}}})$ with $\alpha' = \min\{1, K\alpha\}$.

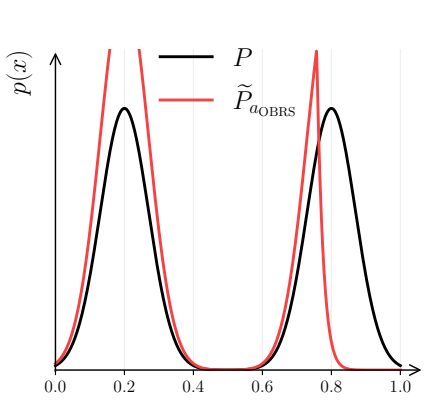


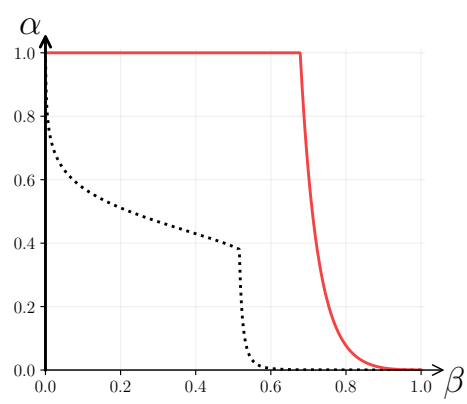
Acceptance rate: 55.54%

Proposition 3.3 (Precision and Recall Improvement)

Let $K \leq M$ be the budget for the OBRS. For any $(\alpha, \beta) \in \mathrm{PRD}(P, \widehat{P})$ we have $(\alpha', \beta) \in \mathrm{PRD}(P, \widetilde{P}_{a_{\mathrm{OBRS}}})$ with $\alpha' = \min\{1, K\alpha\}$.



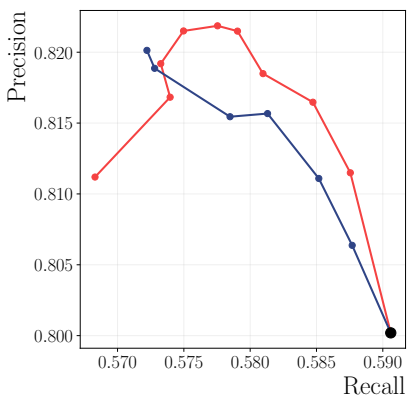


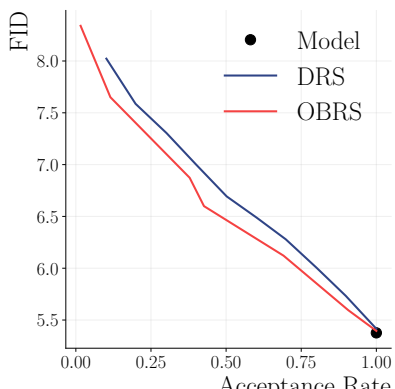


Acceptance rate: $3.64.10^{-3}$ %

Improving Precision and Recall

In Practice





GAN on CelebA

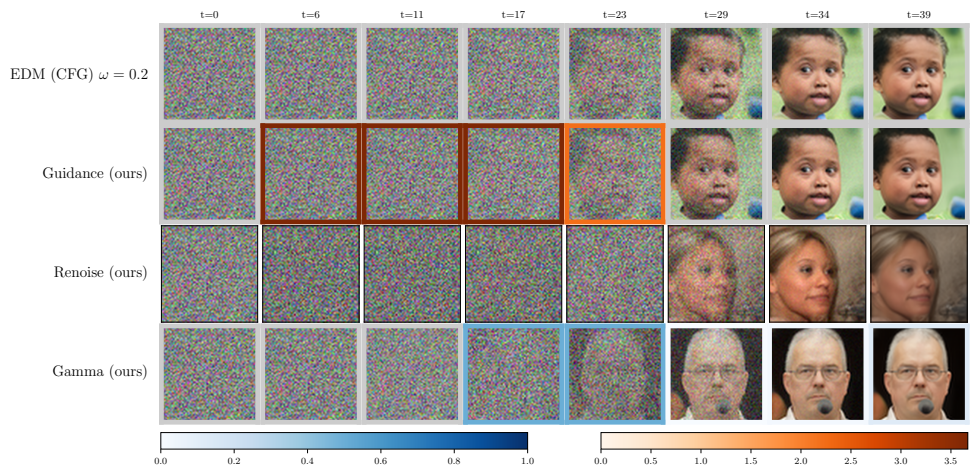
1/K	FID	P	R
0.25	1.57	78.48	86.73
0.50	1.58	78.23	86.05
0.75	1.77	77.94	86.54
1	1.97	77.91	86.62

Diffusion model on CIFAR-10

Vérine et al. · AISTATS 2024

Same denoiser, different sampling trajectories

Inference-time control in diffusion models



Keep the denoiser weights fixed. Learn *when* to apply guidance, inject stochasticity or revisit an earlier noise level.

Learn the sampler instead of enlarging the model

Another lever on the quality–diversity trade-off

What is learned?

Current sample and noise level

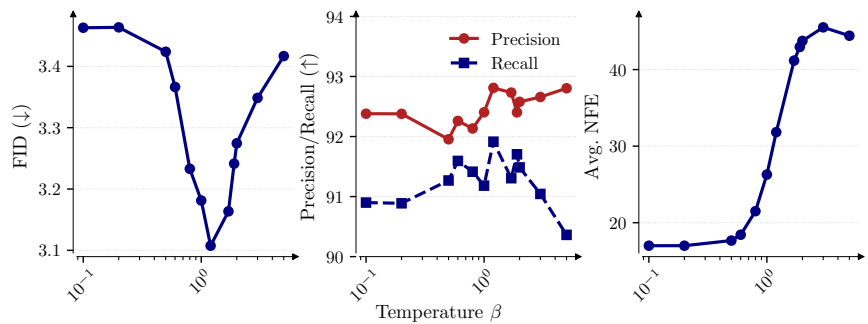
↓

Sampling-control policy

↓

Next denoising step

The policy changes; the denoiser does not. This is a diffusion example, not a GAN algorithm to copy into A2.



CIFAR-10: temperature of the learned renoising policy changes precision, recall and the number of function evaluations (NFE).

Boosting Generative models:



Figure. Left: Samples from the dataset given high weights by the discriminator. Right: Samples from the dataset given low weights by the discriminator. The next model will focus on the sample on the right. Source: Tolstikhin et al. [27]

- Tolstikhin et al. [27]
- Grover and Ermon [14]

Other Methods to improve Precision and Recall

Gradient Ascent

Using the discriminator as a classifier and perform a gradient descent:

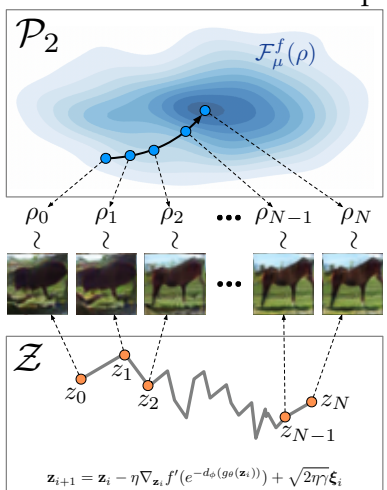


Figure. Source: Ansari et al. [1]

- Ansari et al. [1]
- Tanaka [26]
- Che et al. [9]

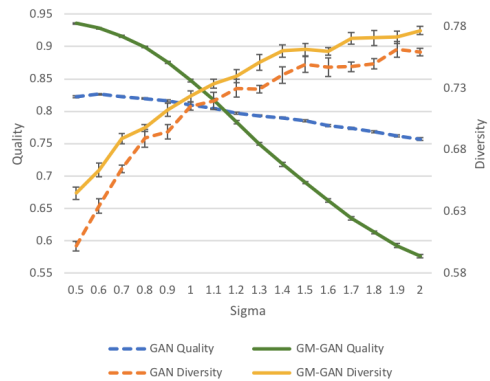
Other Methods to improve Precision and Recall

Gaussian Mixtures

Training a Gaussian Mixture $\mathcal{N}(\mu_k, \sigma I)$ in the latent space:



(a) $\sigma = 0.1$ (b) $\sigma = 1$ (c) $\sigma = 2$



(d) Precision and Recall

Figure. Source: Ben-Yosef and Weinshall [4]

- Ben-Yosef and Weinshall [4]
- Pandeva and Schubert [21]
- Alternative idea: Use Expectation-Maximization Bishop [5]

Assignment 2

Generating MNIST with a limited-capacity model

Assignment 2: train a GAN on MNIST

One baseline, a clear research question

1. Train the provided generator on MNIST.
2. Choose one or two motivated improvements.
3. Measure and explain their effect on quality and diversity.

The generator architecture is fixed.

You train its weights; you do not make it bigger.



▸ Starter repository: DataLabAssignement2

Architecture reminder

A constraint of the assignment

Do not change the generator architecture.

- Keep the provided layers, widths and latent/output dimensions.
- Train the weights; investigate preprocessing, objectives and sampling.
- A larger generator or a replacement CNN is outside the assignment.

Why is the generator architecture fixed?

Research beyond a larger model

- Architecture design is a valid research direction, especially for complex tasks.
- On MNIST, a large CNN can hide the questions we want to investigate.
- With limited capacity, can we use the data, training objective and sampling strategy more effectively?
- A shared architecture makes interventions easier to compare and interpret.

Fixed architecture $\neq$ frozen weights

Keep the provided generator structure and input/output dimensions. Train its parameters and study the surrounding pipeline.

Three work packages, one research project

A possible division of work for a group of three

Data

Selection, preprocessing, augmentation.

What distribution does the model learn?

Training

Losses, regularisation, optimisation.

How does the model use its capacity?

Inference

Truncation, rejection, latent refinement.

How do we sample from the trained model?

Share a baseline, checkpoints, interfaces and evaluation protocol.

Compare each intervention separately, then useful combinations.

This is an option for working in parallel, not a requirement to implement three unrelated methods or to present three mini-projects.

Reading ideas: Preprocessing

Selecting and augmenting the training data

Paper / method	Main idea to investigate
Instance Selection for GANs	Select training examples; measure the effect on coverage.
DiffAugment	Apply differentiable augmentations to real and generated images.
Adaptive Discriminator Augmentation (ADA)	Adapt augmentation strength to discriminator overfitting.

Keep transformations meaningful for MNIST: a transformed image must still represent the intended digit.

Reading ideas: Preprocessing

Other ways to construct the discriminator inputs

Paper / method	Main idea to investigate
Deceive D / Adaptive Pseudo Augmentation	Adaptively mix generated images into the real-data distribution.
Diffusion-GAN	Use diffusion-based noise augmentation during GAN training.
AmbientGAN	Advanced: learn from noisy or incomplete measurements.

These methods also change the training protocol. AmbientGAN is an optional research extension using a declared measurement model, not a change to the evaluation data.

Reading ideas: Training

Objectives and divergences

Paper / method	Main idea to investigate
LSGAN	Replace cross-entropy with a least-squares objective.
f-GAN	Choose the divergence used to train the generator.
Wasserstein GAN	Use a Wasserstein critic with a Lipschitz constraint.
Improved WGAN / WGAN-GP	Replace weight clipping with a gradient penalty.
PR-divergence	Target a particular precision–recall trade-off.

If you choose WGAN or WGAN-GP, keep the textbook explanation short. Your experiments and interpretation are the main contribution.

Reading ideas: Training

Regularisation and discriminator geometry

Paper / method	Main idea to investigate
Spectral Normalization for GANs	Control discriminator layer norms.
DRAGAN: On Convergence and Stability of GANs	Regularise gradients near real data.
Which Training Methods for GANs Converge?	Investigate zero-centred penalties such as R1.
Consistency Regularization for GANs	Make discriminator predictions consistent under augmentation.
Improved Consistency Regularization	Study balanced and latent consistency regularisation.

Adapt the discriminator or its objective when the method requires it. The generator architecture remains fixed.

Reading ideas: Training

Optimisation and alternative discriminator objectives

Paper / method	Main idea to investigate
TTUR	Use different learning rates for generator and discriminator.
SAN: Slicing Adversarial Network	Change the normalised discriminator layer and its objective.
The Relativistic Discriminator	Compare real and generated samples relatively.
Unrolled GANs	Differentiate through several discriminator updates.
Improved Techniques for Training GANs	Investigate feature matching or one-sided label smoothing.

Compare at a stated compute budget. Unrolling, extra gradients and additional updates are not free.

Reading ideas: Inference

Truncation and sampling correction

Paper / method	Main idea to investigate
BigGAN / latent truncation	Change the latent sampling distribution; measure lost coverage.
Discriminator Rejection Sampling	Reject poor samples using the discriminator.
MH-GAN	Wrap the generator in a Metropolis–Hastings chain.
Optimal Budgeted Rejection Sampling	Optimise rejection subject to a sampling budget.

Keep the trained generator. Report acceptance rates, total generation time and the full evaluation batch.

Reading ideas: Inference

Latent reweighting and sample refinement

Paper / method	Main idea to investigate
Discriminator Driven Latent Sampling	Use Langevin dynamics to refine latent variables.
Discriminator Gradient Flow	Refine generated samples through discriminator gradients.
Discriminator Optimal Transport	Use the discriminator to guide sample transport.
Latent Reweighting	Learn latent weights for rejection or latent gradient ascent.
MaGNET	Sample more uniformly on the learned manifold.

Manifold uniformity is not the same as semantic class balance. Check FID, precision, recall and computation cost.

A clean experimental protocol

What changed, and why did it help?

1. Reproduce a baseline and record configuration, seed and checkpoint.
2. State a hypothesis and change one component at a time.
3. Keep the evaluation protocol and computational budgets comparable.
4. Run ablations; distinguish training cost from generation cost.
5. Show representative samples and metrics, including informative failures.
6. Interpret the trade-off: what improved, what degraded, and why?

If possible, repeat runs to distinguish an improvement from seed variability.

Repository and generation contract

A Python project that can be rerun

Keep the starter interface

`generate.py` loads your trained generator from `checkpoints/` and produces exactly **10 000** MNIST images, **28** $\times$ **28**, under `samples/`.

```
python generate.py --batch_size 256
```

- Document training and generation in the README.
- Declare dependencies in `requirements.txt`.
- Make the checkpoint available to the evaluation script.

Python code should be clean and reproducible. Code quality itself is not graded; your results and claims must be verifiable.

Evaluation and lessons from A1

A focused analysis and a clear presentation

Experiments and report

FID, precision, recall and visual inspection.

Explain what the measurements mean. The theoretical PR curve and the finite-sample image estimator are different objects.

Focus on your experiments and interpretation, not generic GAN background.

Five minutes means five minutes

One method explained well beats a catalogue.

Use 16:9 slides, readable plots and properly typeset equations.

Rehearse, adapt the slide count, speak loudly and address the room.

AI tools are allowed. You must verify and defend the code, results and writing.

A2 DELIVERABLES AND DATES

ALL TIMES: EUROPE/PARIS

	Preliminary groups	Final groups
Presentation	Thursday 5 November	Thursday 12 November
`slides.pdf`	4 November, 23:59	11 November, 23:59

For all groups, including preliminary groups:

- `report.pdf`: **11 November, 23:59**.
- Final code changes: **12 November, 05:00**.

Presentation: **5 minutes**, alarm at 5 minutes, then questions/transition.

Report: **5 pages of content maximum** plus a group cover page. Optional appendices are not read. Required filenames and deadlines; penalties apply.

▸ Course website: assignment links and updates

A2: PRELIMINARY OR FINAL PRESENTATION?

VOLUNTEERS FIRST, THEN RANDOM ALLOCATION

Would you like to present in preliminary on 5 November?

Tell me by the end of today, **Thursday 22 October**. Include your group name in the email.

alexandre.verine@dauphine.psl.eu

- On **Saturday 31 October**, the remaining presentation dates will be assigned randomly.
- Remaining students without a complete group will also be assigned then.
- Each group presents once: preliminary on **5 November** or final on **12 November**.

A few tips for your presentation

Focus on your contribution

- If you choose WGAN or Improved WGAN, do not spend too much time explaining the standard method. Focus on your experiments.
- Choose **one or two methods**, no more.
- Reduce the number of slides: you have **five minutes**.
- Explain what changed, show the evidence, and interpret the results.

Thanks !

References I

[1] Ansari, A. F., Ang, M. L., and Soh, H. (2021). Refining Deep Generative Models via Discriminator Gradient Flow. arXiv:2012.00780 [cs, stat].

[2] Azadi, S., Olsson, C., Darrell, T., Goodfellow, I., and Odena, A. (2019). Discriminator Rejection Sampling. arXiv:1810.06758 [cs, stat].

[3] Barratt, S. and Sharma, R. (2018). A Note on the Inception Score. arXiv:1801.01973 [cs, stat].

[4] Ben-Yosef, M. and Weinshall, D. (2018). Gaussian Mixture Generative Adversarial Networks for Diverse Datasets, and the Unsupervised Clustering of Images. *arXiv:1808.10356 [cs, stat]*. arXiv: 1808.10356.

[5] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag.

[6] Borji, A. (2022). Pros and cons of GAN evaluation measures: New developments. *Computer Vision and Image Understanding*, 215:103329.

[7] Brock, A., Donahue, J., and Simonyan, K. (2019). Large Scale GAN Training for High Fidelity Natural Image Synthesis. arXiv:1809.11096 [cs, stat].

[8] Bronnec, F. L., Verine, A., Negrevergne, B., Chevaleyre, Y., and Allauzen, A. (2024). Exploring Precision and Recall to assess the quality and diversity of LLMs. *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics*. arXiv:2402.10693 [cs].

References II

[9] Che, T., Zhang, R., Sohl-Dickstein, J., Larochelle, H., Paull, L., Cao, Y., and Bengio, Y. (2020). Your GAN is Secretly an Energy-based Model and You Should Use Discriminator Driven Latent Sampling. In *Advances in Neural Information Processing Systems*, volume 33, pages 12275–12287. Curran Associates, Inc.

[10] Chong, M. J. and Forsyth, D. (2020). Effectively Unbiased FID and Inception Score and where to find them. arXiv:1911.07023 [cs].

[11] Frolov, S., Hinz, T., Raue, F., Hees, J., and Dengel, A. (2021). Adversarial text-to-image synthesis: A review. *Neural Networks*, 144:187–209.

[12] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Networks. In *27th Conference on Neural Information Processing Systems (NeurIPS 2014)*. arXiv: 1406.2661.

[13] Grover, A., Dhar, M., and Ermon, S. (2018). Flow-GAN: Combining Maximum Likelihood and Adversarial Learning in Generative Models. arXiv:1705.08868 [cs, stat].

[14] Grover, A. and Ermon, S. (2017). Boosted Generative Models. arXiv:1702.08484 [cs, stat].

[15] Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

References III

[16] Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., and Aila, T. (2020). Analyzing and Improving the Image Quality of StyleGAN. arXiv:1912.04958 [cs, eess, stat].

[17] Kingma, D. P. and Dhariwal, P. (2018). Glow: Generative Flow with Invertible 1x1 Convolutions. In *32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.*, volume 31.

[18] Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., and Aila, T. (2019). Improved Precision and Recall Metric for Assessing Generative Models. In *33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.* arXiv: 1904.06991.

[19] Nguyen, X., Wainwright, M. J., and Jordan, M. I. (2009). On surrogate loss functions and f-divergences. *The Annals of Statistics*, 37(2). arXiv:math/0510521.

[20] Nowozin, S., Cseke, B., and Tomioka, R. (2016). f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization. arXiv:1606.00709 [cs, stat].

[21] Pandeva, T. and Schubert, M. (2019). MMGAN: Generative Adversarial Networks for Multi-Modal Distributions. *arXiv:1911.06663 [cs, stat]*. arXiv: 1911.06663.

[22] Sajjadi, M. S. M., Bachem, O., Lucic, M., Bousquet, O., and Gelly, S. (2018). Assessing Generative Models via Precision and Recall. In *32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada*. arXiv: 1806.00035.

References IV

[23] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved Techniques for Training GANs. arXiv:1606.03498 [cs].

[24] Simon, L., Webster, R., and Rabin, J. (2019). Revisiting precision recall definition for generative modeling. In *Proceedings of the 36th International Conference on Machine Learning*, pages 5799–5808. PMLR. ISSN: 2640-3498.

[25] Siry, R., Webster, R., Simon, L., and Rabin, J. (2023). On the Theoretical Equivalence of Several Trade-Off Curves Assessing Statistical Proximity. *Journal of Machine Learning Research*, 24.

[26] Tanaka, A. (2019). Discriminator optimal transport. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

[27] Tolstikhin, I., Gelly, S., Bousquet, O., Simon-Gabriel, C.-J., and Schölkopf, B. (2017). AdaGAN: Boosting Generative Models. arXiv:1701.02386 [cs, stat].

[28] Verine, A., Bronnec, F. L., Zheng, K., Allauzen, A., Chevaleyre, Y., and Negrevergne, B. (2025). Improving Diversity in Language Models: When Temperature Fails, Change the Loss. ICML 2025. arXiv. arXiv:2508.09654 [cs].

[29] Yann LeCun, Corinna Cortes, and Burges, C. (2010). MNIST handwritten digit database. *ATT Labs*, 2.