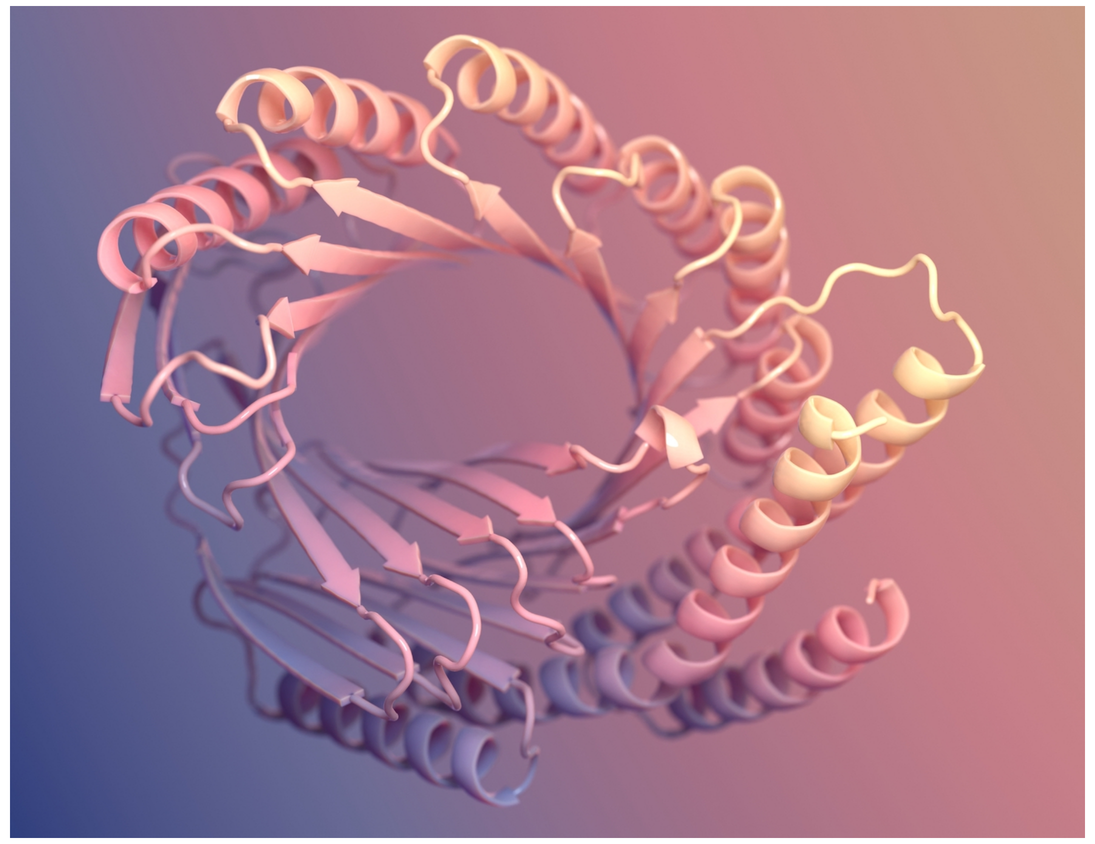
Guiding Deep Probabilistic Models

Timur Garipov

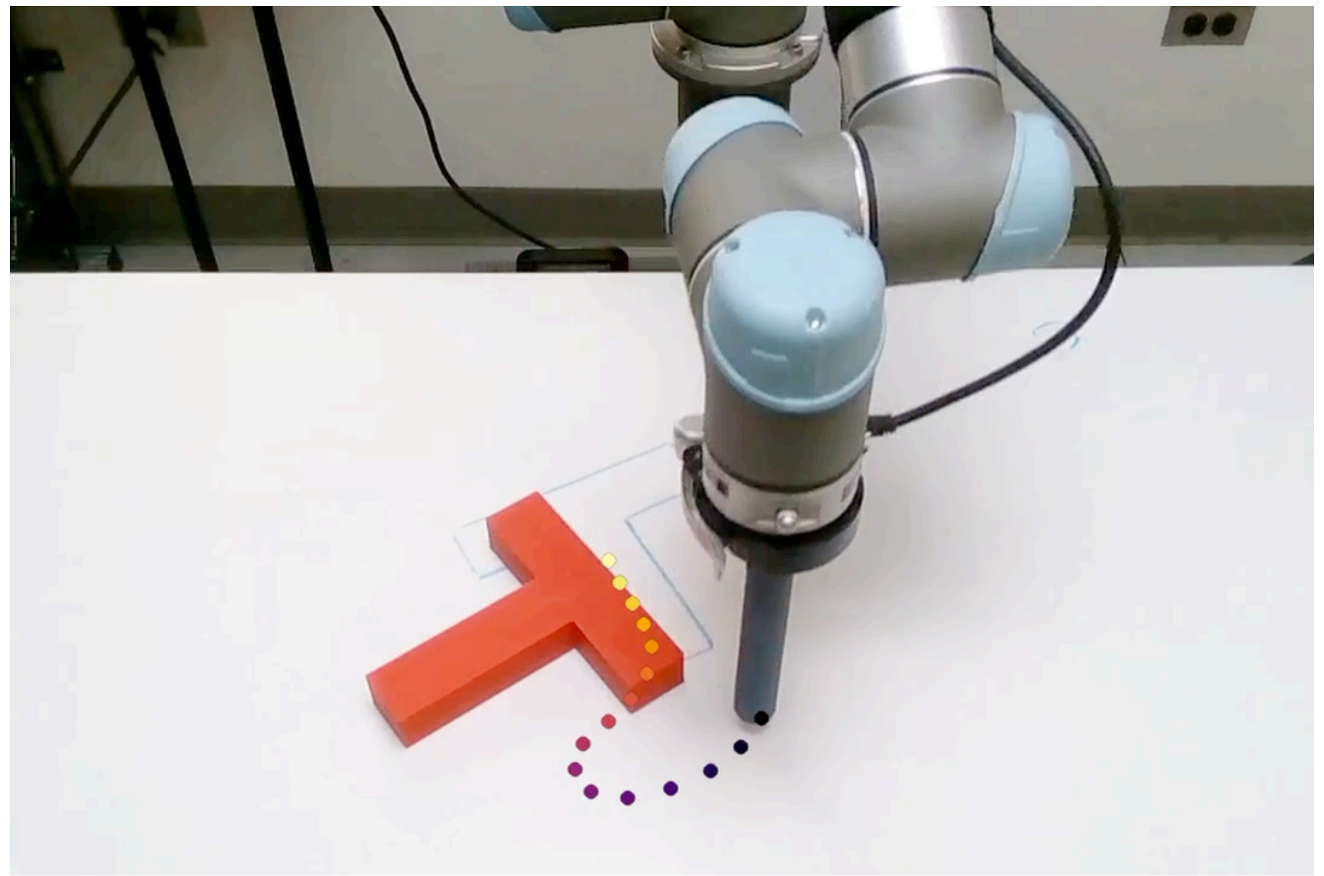
06/13/2024

GPT-4
[OpenAI, 2023]

DALL-E 3
[OpenAI, 2023]



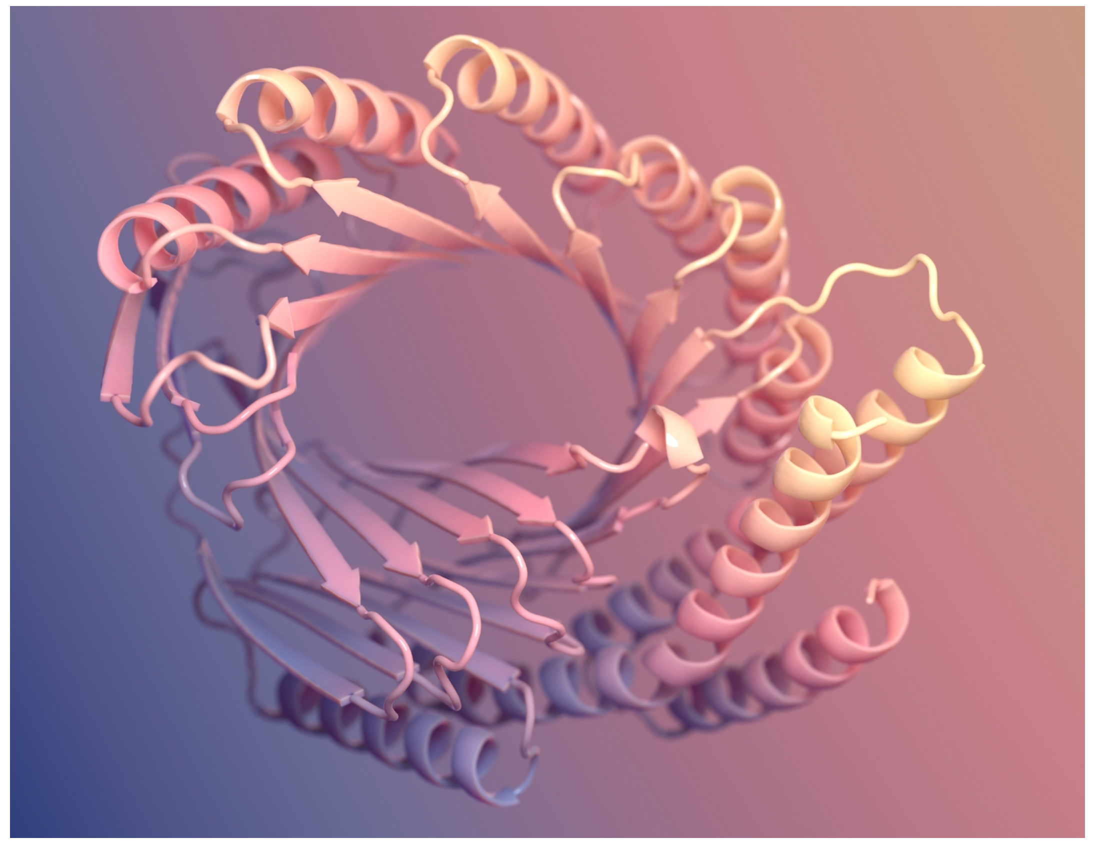
RFdiffusion
[Watson et al., 2023]



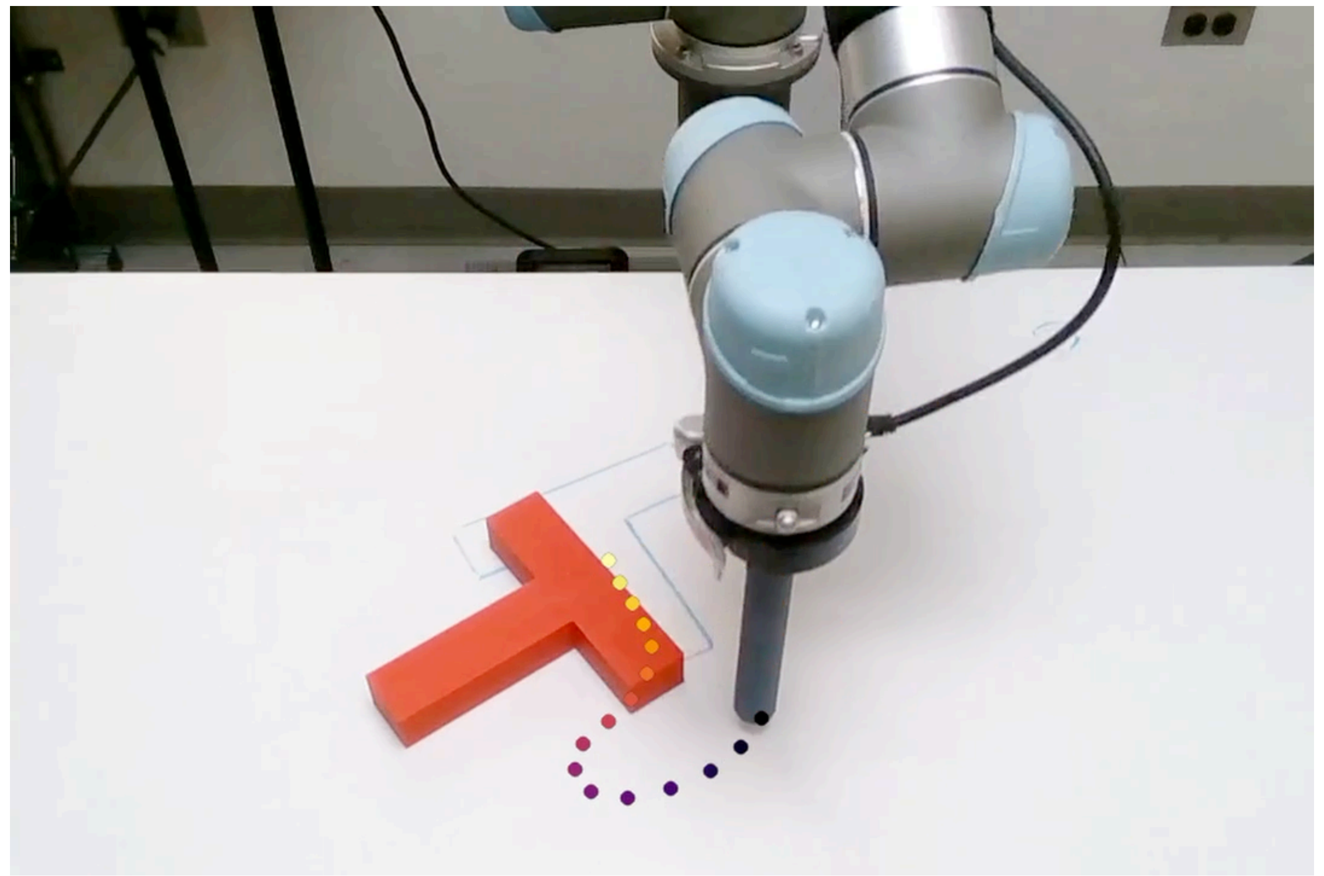
Diffusion Policy
[Chi et al., 2023]

GPT-4
[OpenAI, 2023]

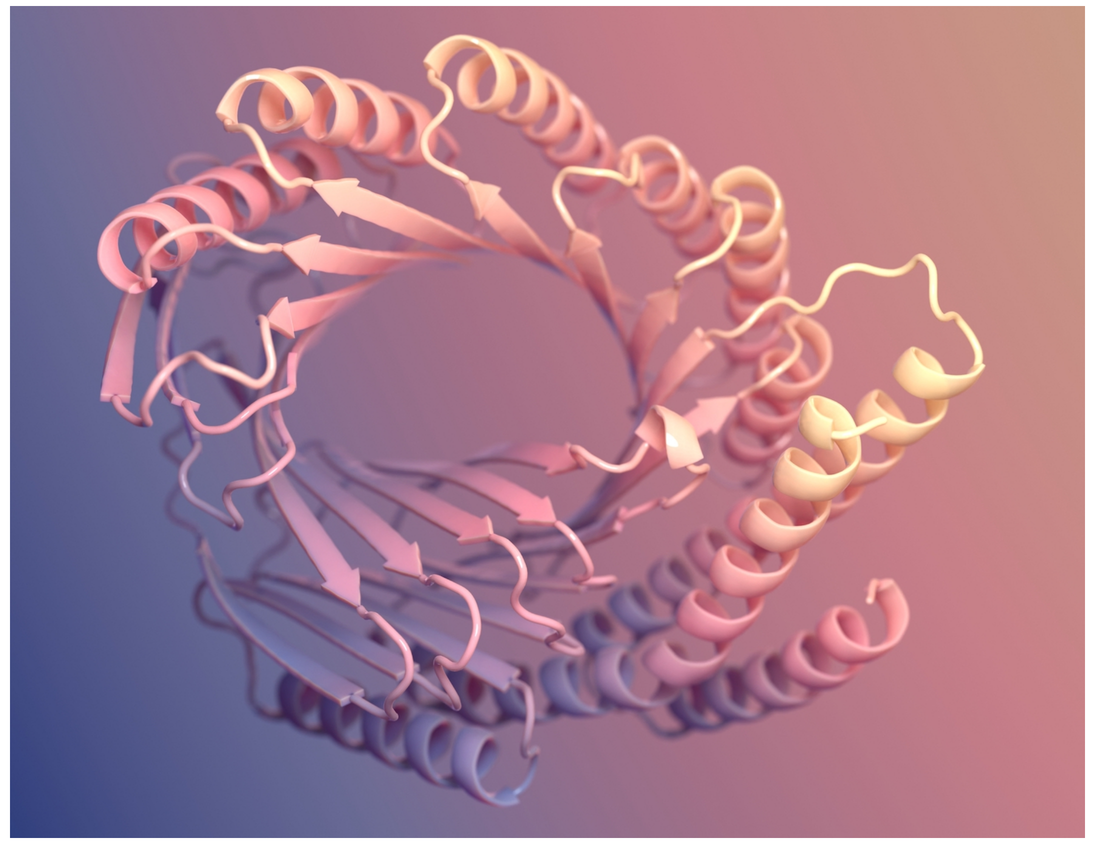
DALL-E 3
[OpenAI, 2023]

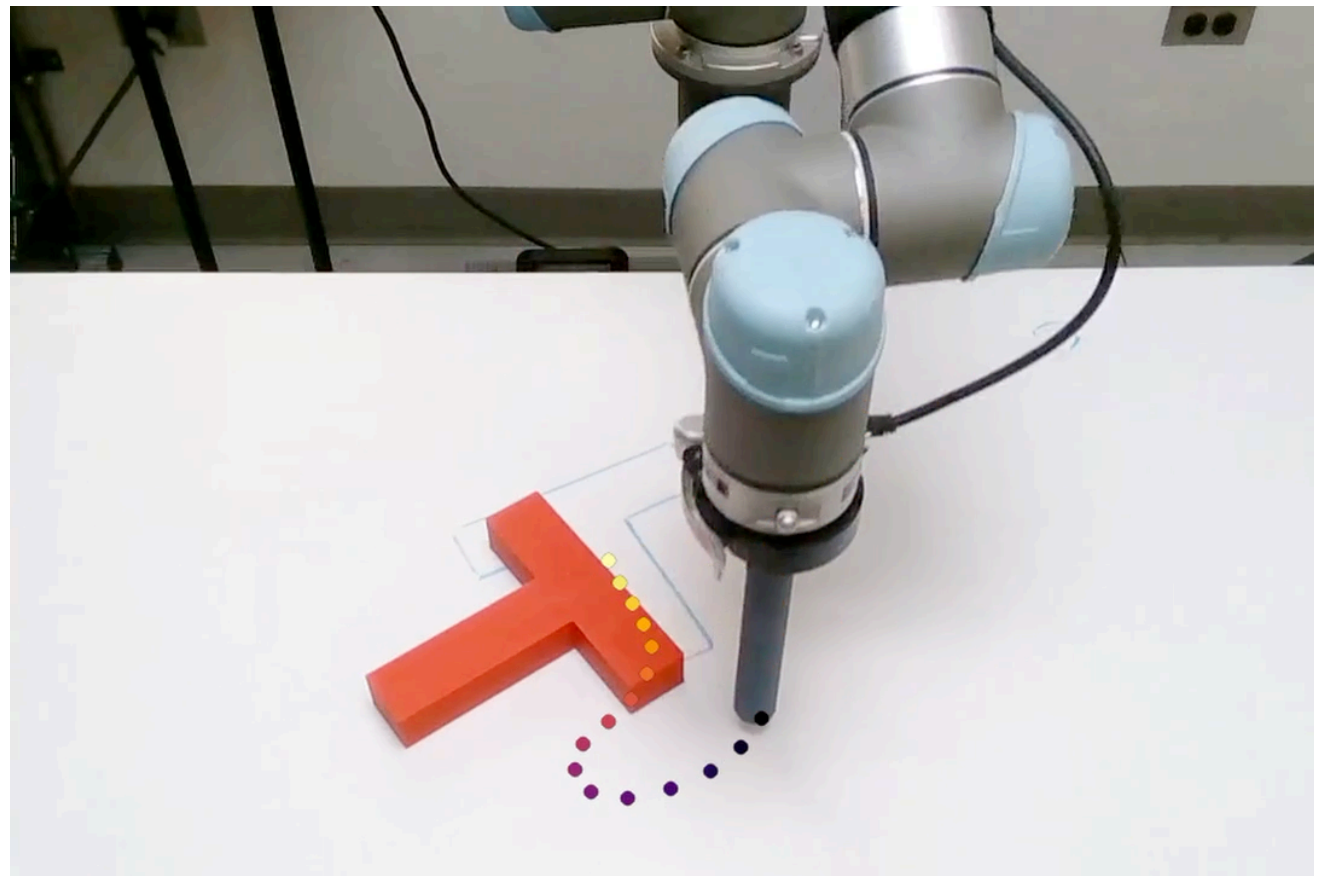


RFdiffusion
[Watson et al., 2023]



Diffusion Policy
[Chi et al., 2023]





GPT-4
[OpenAI, 2023]

DALL-E 3
[OpenAI, 2023]

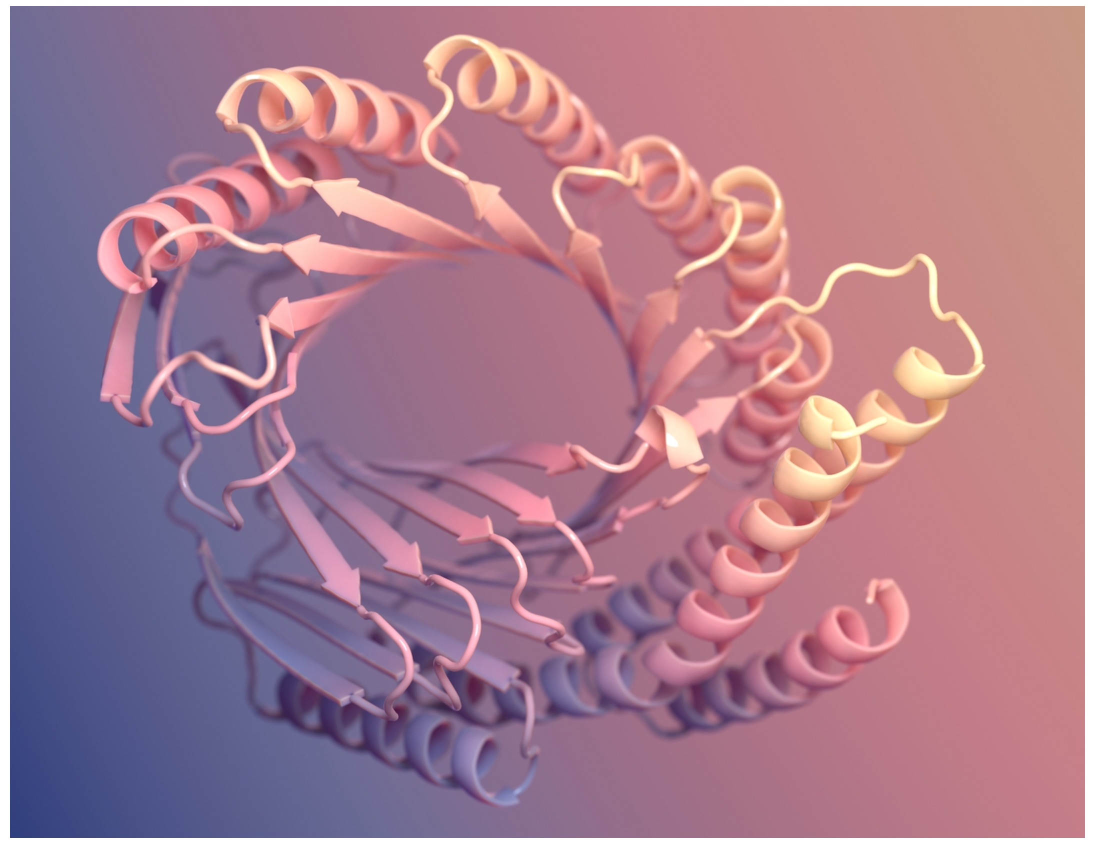
RFdiffusion
[Watson et al., 2023]

Diffusion Policy
[Chi et al., 2023]

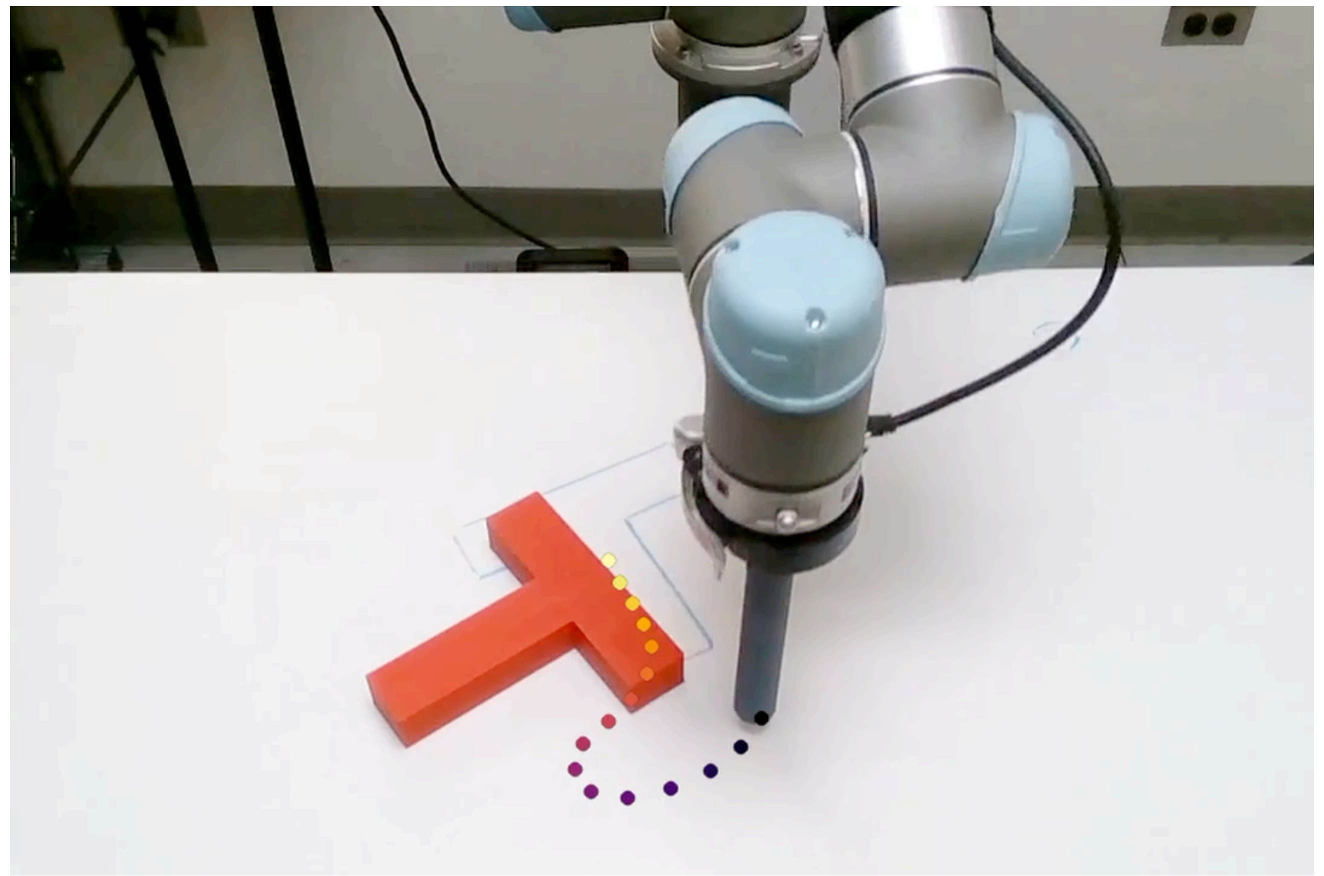
Ground-breaking impact in language modeling, image generation, sciences, robotics

GPT-4
[OpenAI, 2023]

DALL-E 3
[OpenAI, 2023]



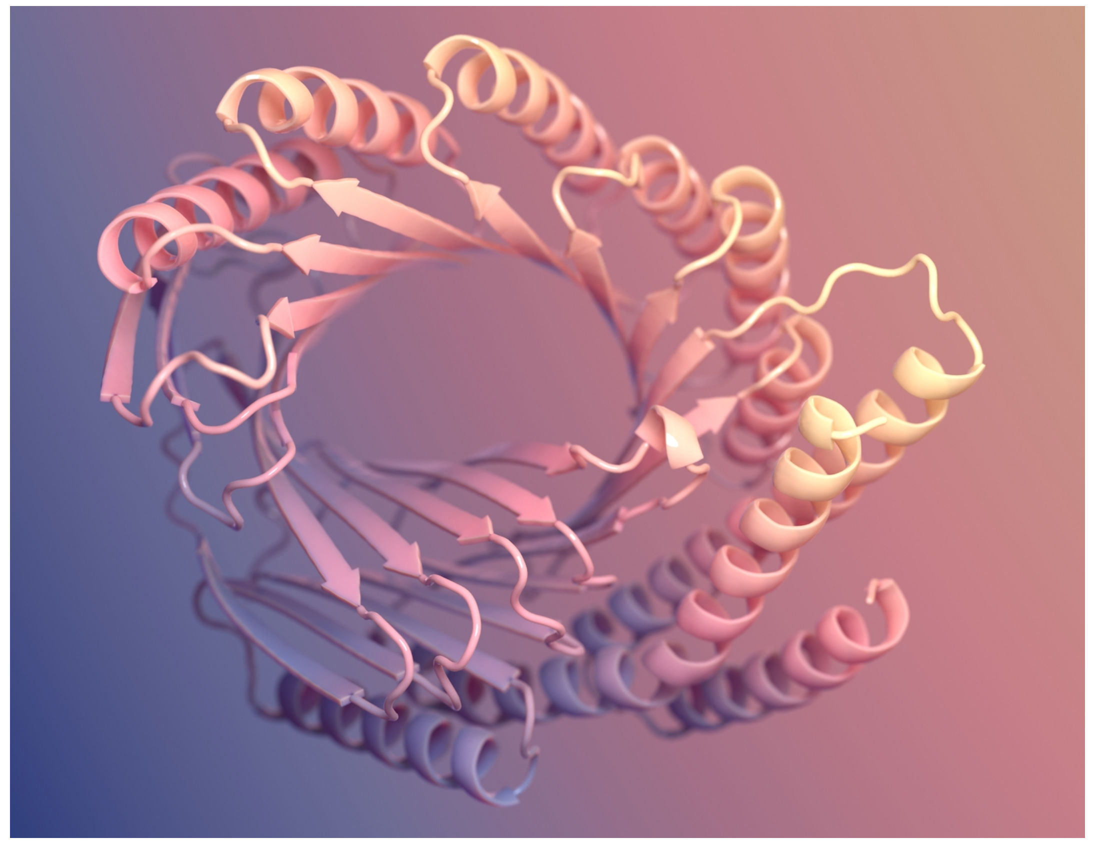
RFdiffusion
[Watson et al., 2023]

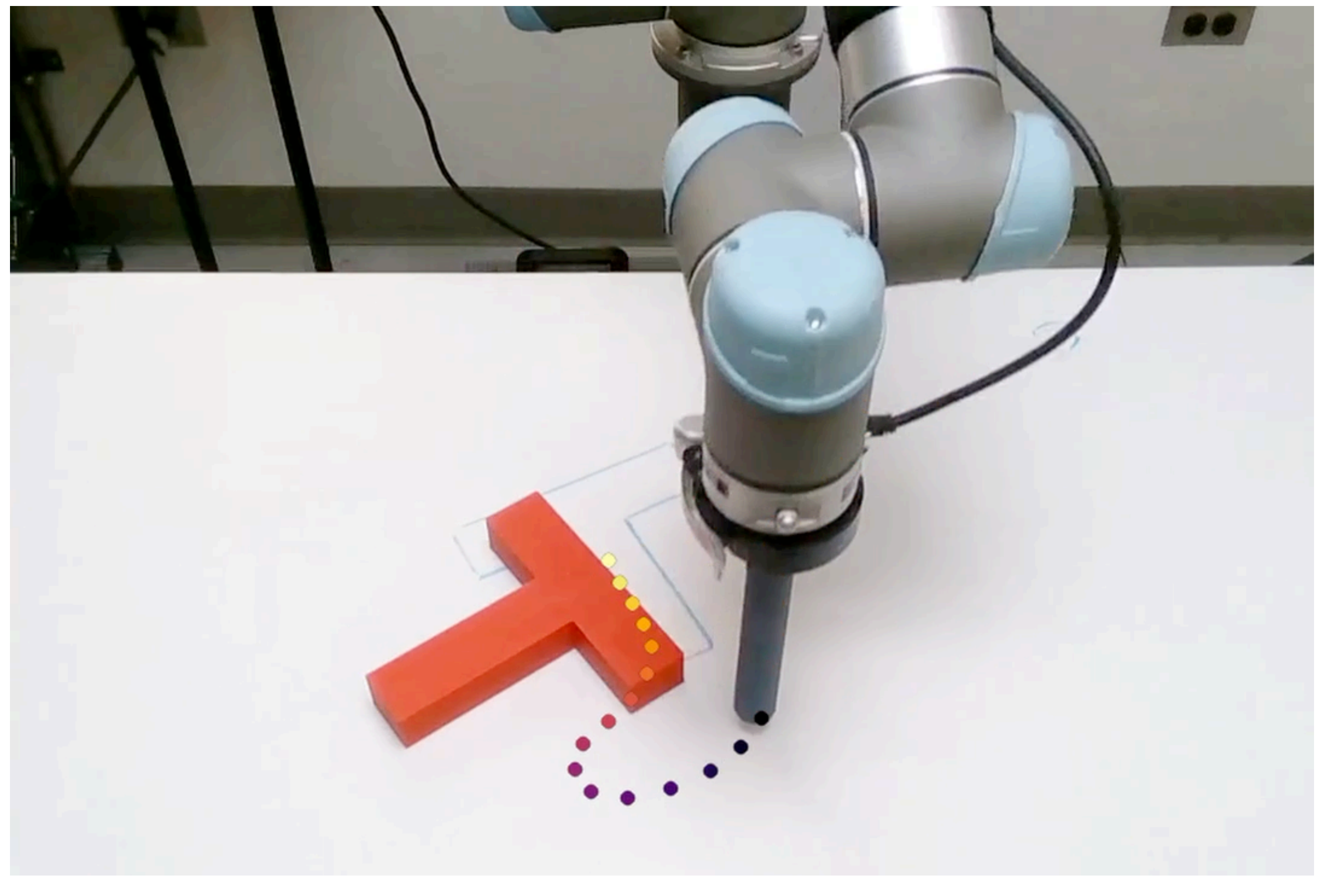


Diffusion Policy
[Chi et al., 2023]

Ground-breaking impact in language modeling, image generation, sciences, robotics

How to make progress in areas where direct supervision signals are limited?





GPT-4
[OpenAI, 2023]

DALL-E 3
[OpenAI, 2023]

RFdiffusion
[Watson et al., 2023]

Diffusion Policy
[Chi et al., 2023]

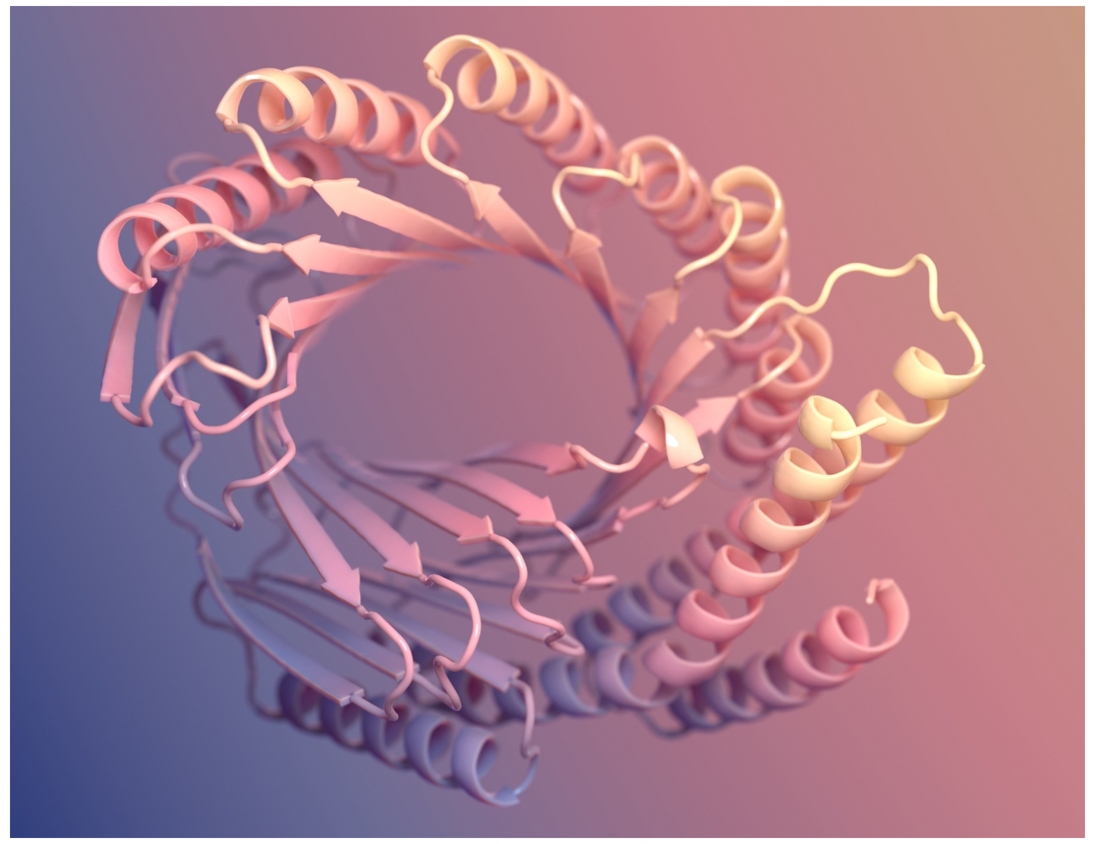
Ground-breaking impact in language modeling, image generation, sciences, robotics

How to make progress in areas where direct supervision signals are limited?

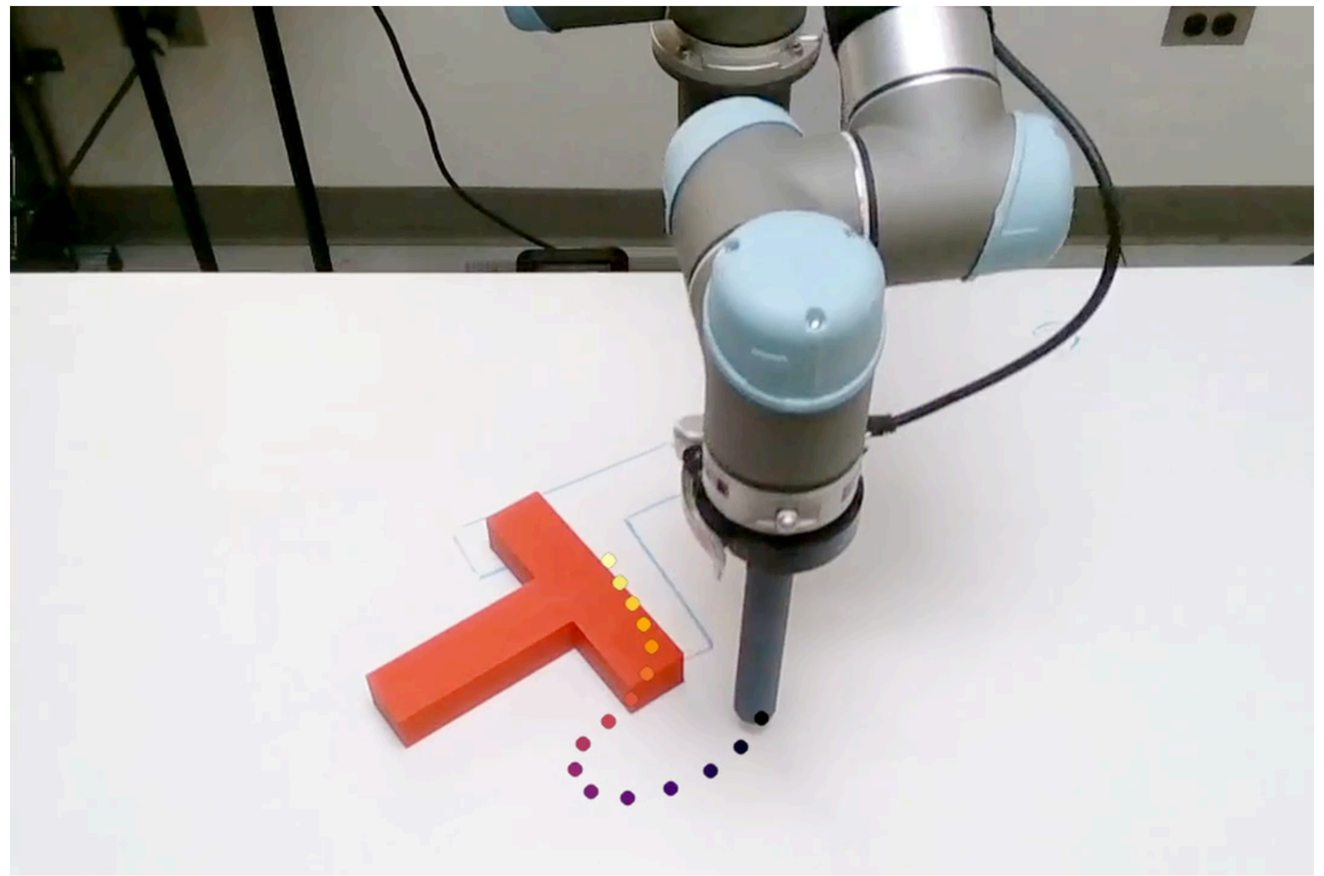
Reasoning and planning

GPT-4
[OpenAI, 2023]

DALL-E 3
[OpenAI, 2023]



RFdiffusion
[Watson et al., 2023]



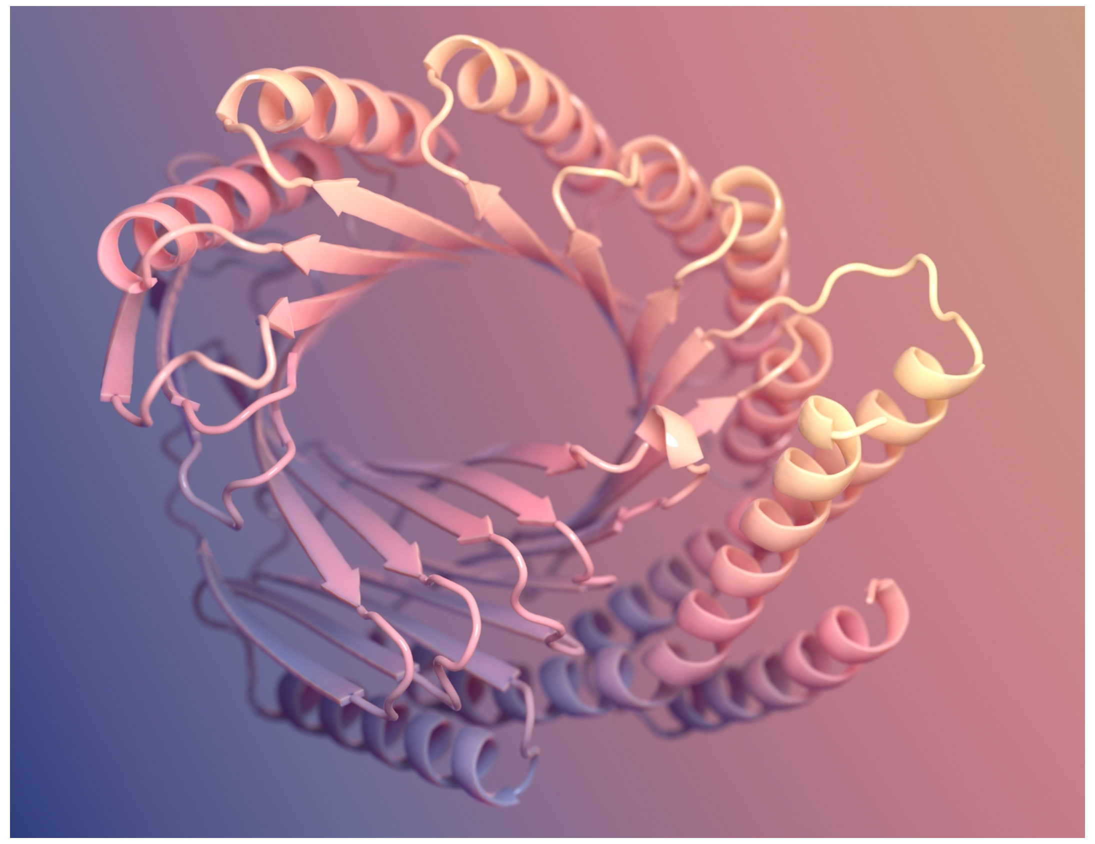
Diffusion Policy
[Chi et al., 2023]

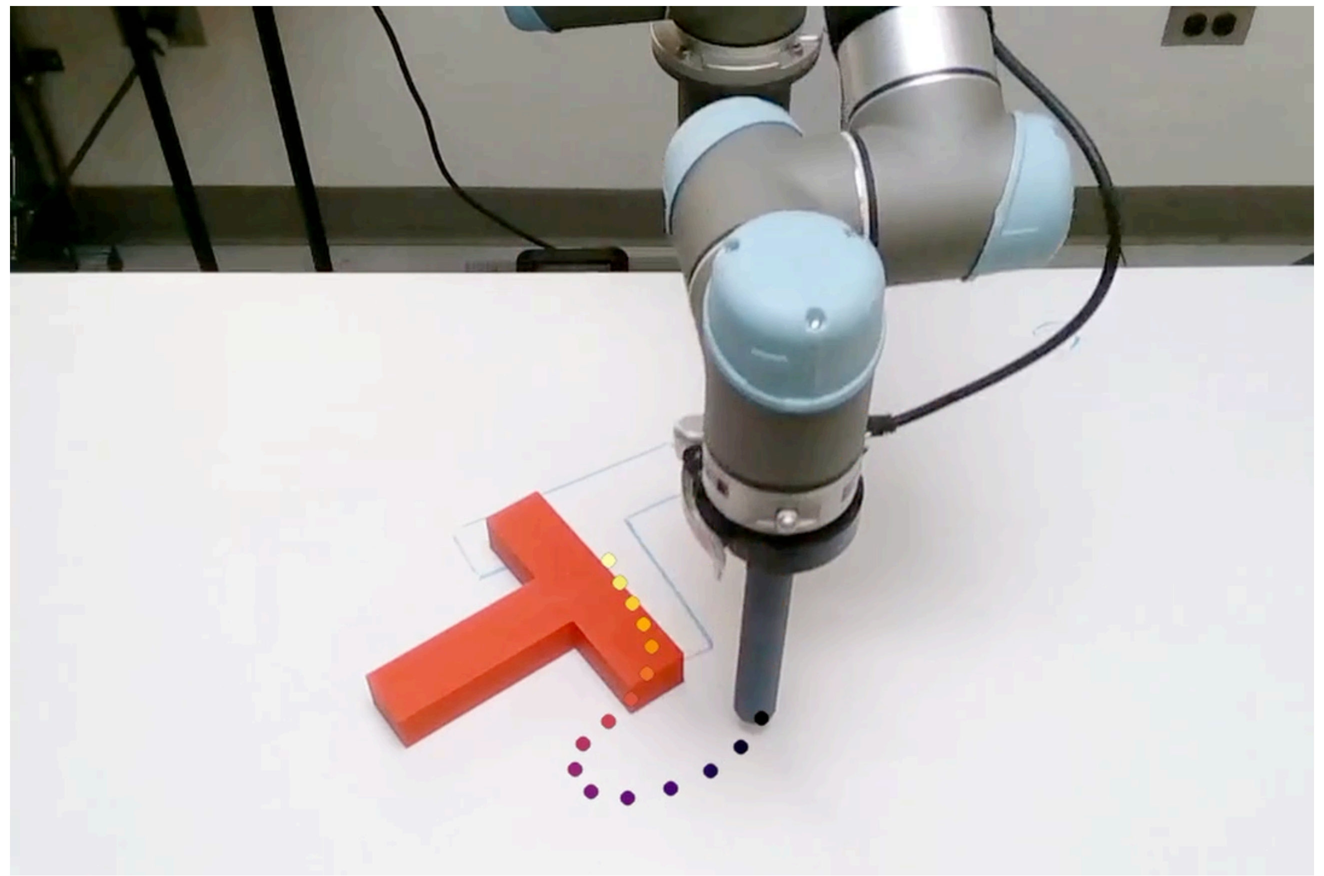
Ground-breaking impact in language modeling, image generation, sciences, robotics

How to make progress in areas where direct supervision signals are limited?

Reasoning and planning

Designing complex experiments and generating hypotheses in science





GPT-4
[OpenAI, 2023]

DALL-E 3
[OpenAI, 2023]

RFdiffusion
[Watson et al., 2023]

Diffusion Policy
[Chi et al., 2023]

Ground-breaking impact in language modeling, image generation, sciences, robotics

How to make progress in areas where direct supervision signals are limited?

Reasoning and planning

Designing complex experiments and generating hypotheses in science

Solving data-scarce and intricately-structured problems

Guidance and Modular Design

Idea: train ML models to provide signal for training / inference in other (larger) models

- Flexible supervision signals
- Controllable inference (generation under multiple constraints)

Guidance and Modular Design

Idea: train ML models to provide signal for training / inference in other (larger) models

- Flexible supervision signals
- Controllable inference (generation under multiple constraints)

Why?

- Need to train and evaluate models without direct supervision signals
- Need to steer multi-step inference processes
- Need to re-use and adapt large pre-trained models
- Need to combine and coordinate multiple models with different areas of expertise

Guidance and Modular Design

Idea: train ML models to provide signal for training / inference in other (larger) models

- Flexible supervision signals
- Controllable inference (generation under multiple constraints)

Why?

- Need to train and evaluate models without direct supervision signals
- Need to steer multi-step inference processes
- Need to re-use and adapt large pre-trained models
- Need to combine and coordinate multiple models with different areas of expertise

Challenges in Guiding Deep Probabilistic Models

- Need to manipulate complex probability distributions in high-dimensional spaces
- Sophisticated and often brittle models, complicated optimization landscapes
- Require computational efficient training and inference algorithms

Research Focus & Goals

Thesis: “Guiding Deep Probabilistic Models”

Research focus areas

- Addressing complex training dynamics between models trained with different objectives
- Design of novel training criteria, addressing shortcomings of existing objectives
- Representing complex probability distributions through generative model combination

Goals

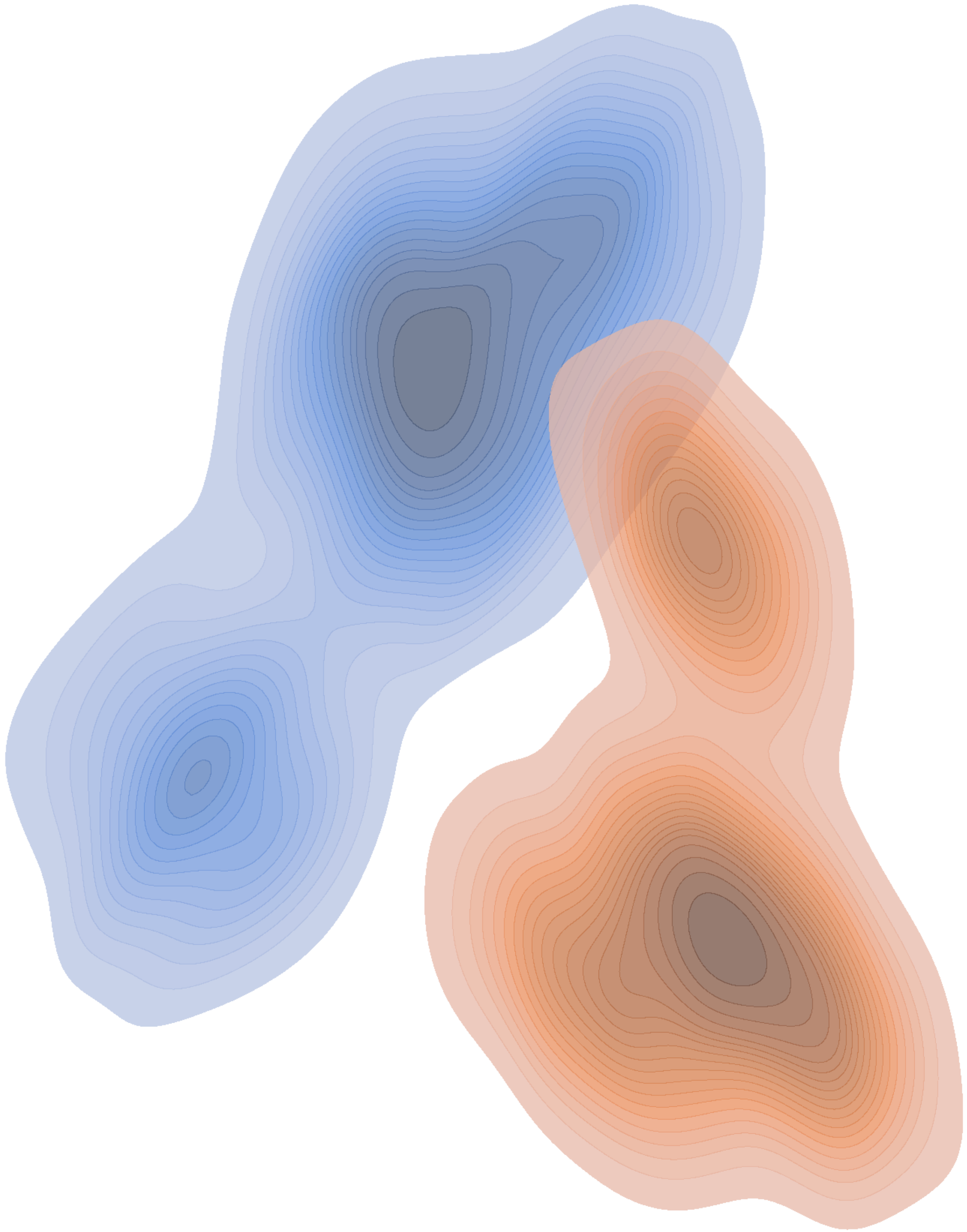
- Novel principled algorithms for training and inference in deep probabilistic models
- Guarantees
  - Training: optimality of the desired target configurations
  - Inference: sampling from target distributions

Chapter II
Pairwise-Discriminator Objectives for Generative Adversarial Networks

The Benefits of Pairwise Discriminators for Adversarial Training
S. Tong*, **T. Garipov***, T. Jaakkola (arXiv Pre-print, 2020)

Guidance for Generative Model Training

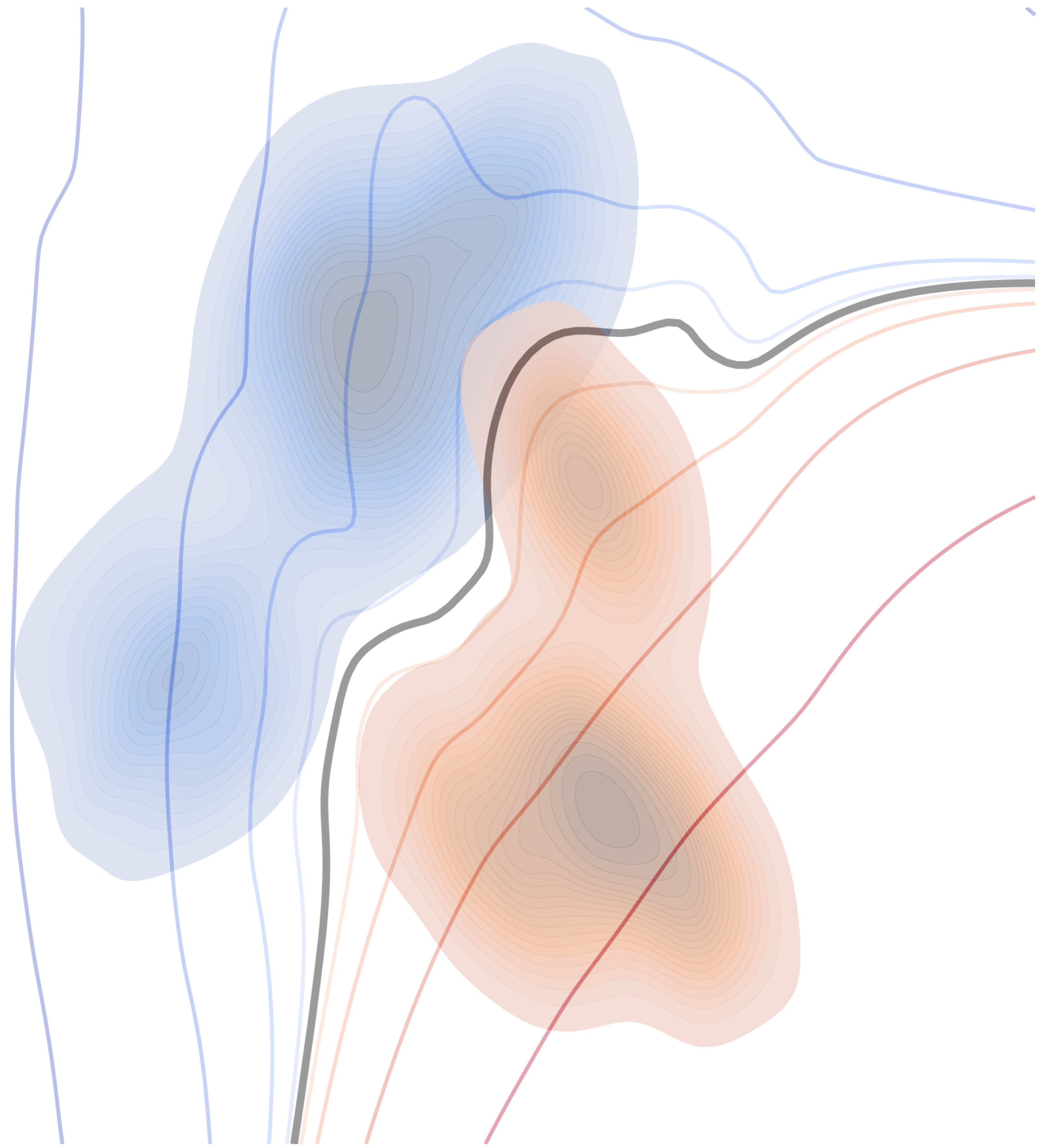
$$\mathrm{JSD}(p \,\|\, q) = \frac{1}{2}\,\mathrm{KL}\left(p \,\middle\|\, \frac{p+q}{2}\right) + \frac{1}{2}\,\mathrm{KL}\left(q \,\middle\|\, \frac{p+q}{2}\right)$$



Guidance for Generative Model Training

$$\mathrm{JSD}(p \,\|\, q) = \frac{1}{2}\,\mathrm{KL}\left(p \,\middle\|\, \frac{p+q}{2}\right) + \frac{1}{2}\,\mathrm{KL}\left(q \,\middle\|\, \frac{p+q}{2}\right)$$

Idea: train a probabilistic classifier to estimate divergence between distributions



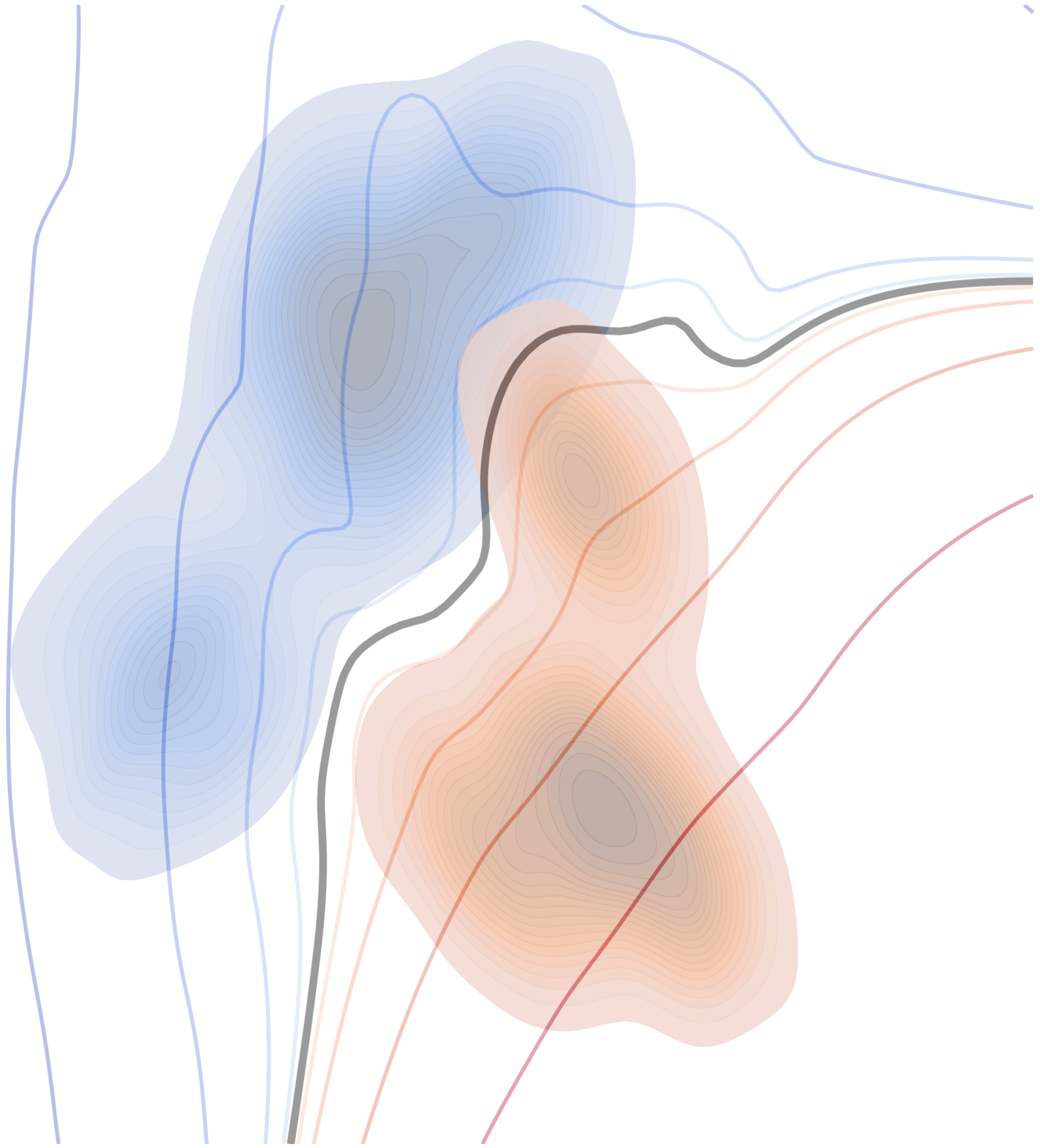
Guidance for Generative Model Training

$$\mathrm{JSD}(p \,\|\, q) = \frac{1}{2}\,\mathrm{KL}\left(p \,\middle\|\, \frac{p+q}{2}\right) + \frac{1}{2}\,\mathrm{KL}\left(q \,\middle\|\, \frac{p+q}{2}\right)$$

$$\mathrm{JSD}(p \,\|\, q) = \log(2) - \frac{1}{2} \min_{u:\mathcal{X}\to\mathbb{R}} \mathcal{L}(p, q, u)$$

$$\mathcal{L}(p, q, u) = \mathbb{E}_{p(x)}\big[\log(1 + \exp(-u(x)))\big] + \mathbb{E}_{q(x)}\big[\log(1 + \exp(u(x)))\big]$$

Idea: train a probabilistic classifier to estimate divergence between distributions



Guidance for Generative Model Training

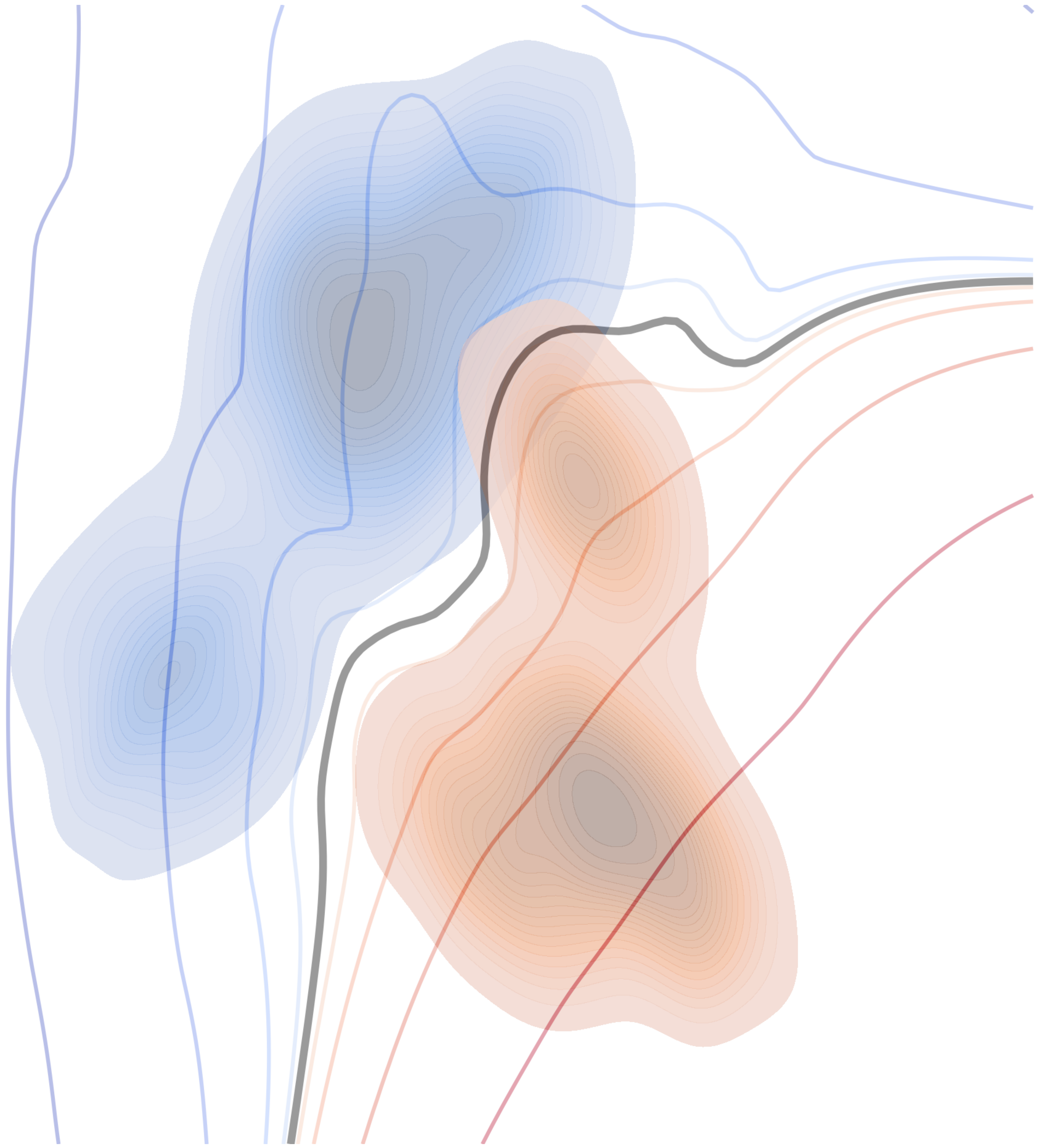
$$\mathrm{JSD}(p\,\|\,q) = \frac{1}{2}\,\mathrm{KL}\left(p \,\middle\|\, \frac{p+q}{2}\right) + \frac{1}{2}\,\mathrm{KL}\left(q \,\middle\|\, \frac{p+q}{2}\right)$$

$$\mathrm{JSD}(p\,\|\,q) = \log(2) - \frac{1}{2}\min_{u:\mathcal{X}\to\mathbb{R}} \mathcal{L}(p,q,u)$$

$$\mathcal{L}(p,q,u) = \mathbb{E}_{p(x)}\big[\log(1+\exp(-u(x)))\big] + \mathbb{E}_{q(x)}\big[\log(1+\exp(u(x)))\big]$$

Generative Adversarial Networks (GANs)

Idea: train a probabilistic classifier to estimate divergence between distributions



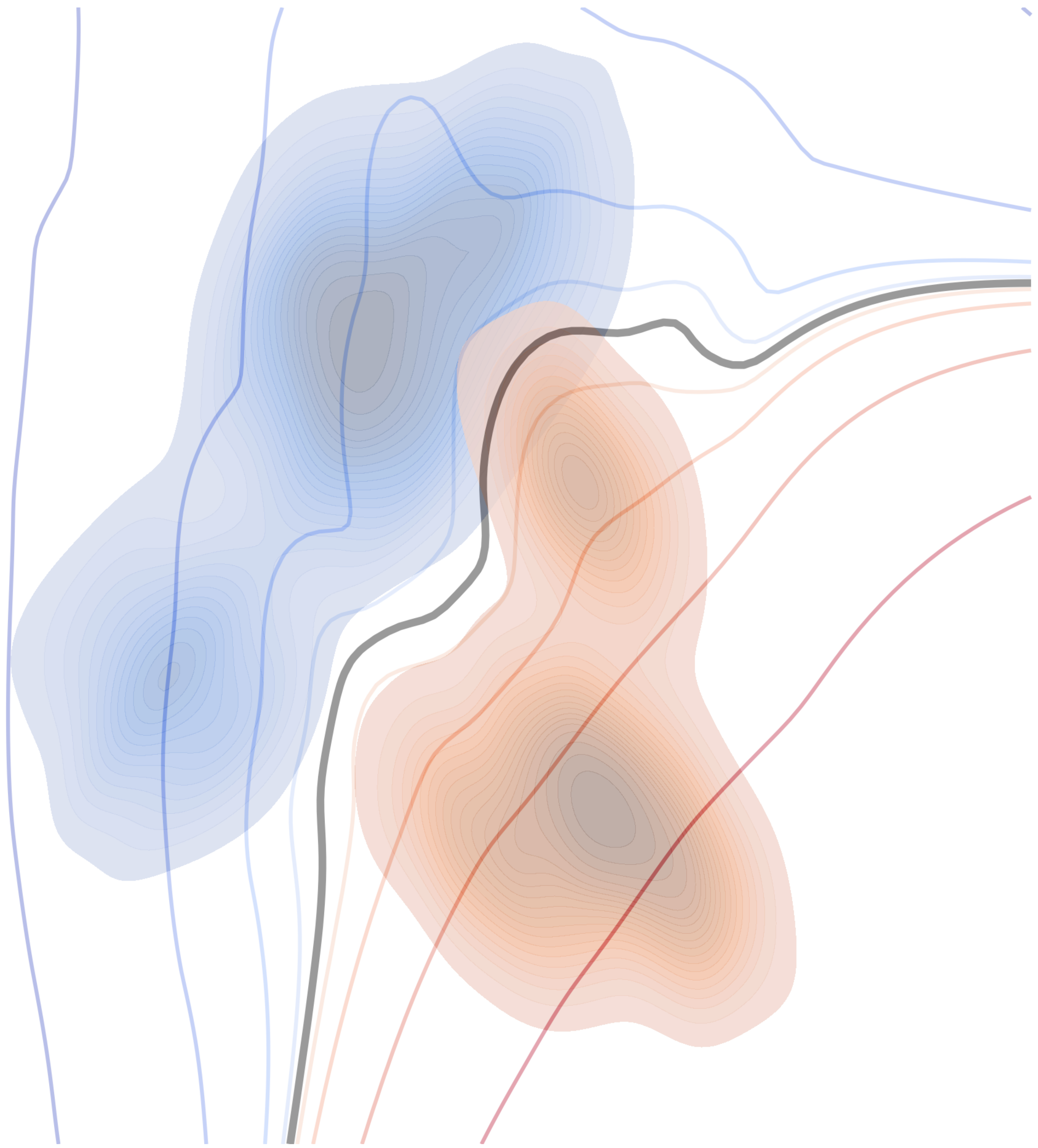
Guidance for Generative Model Training

$$\mathrm{JSD}(p \,\|\, q) = \frac{1}{2}\,\mathrm{KL}\left(p \,\middle\|\, \frac{p+q}{2}\right) + \frac{1}{2}\,\mathrm{KL}\left(q \,\middle\|\, \frac{p+q}{2}\right)$$

$$\mathrm{JSD}(p \,\|\, q) = \log(2) - \frac{1}{2} \min_{u:\mathcal{X}\to\mathbb{R}} \mathcal{L}(p, q, u)$$

$$\mathcal{L}(p, q, u) = \mathbb{E}_{p(x)}\big[\log(1 + \exp(-u(x)))\big] + \mathbb{E}_{q(x)}\big[\log(1 + \exp(u(x)))\big]$$

Idea: train a probabilistic classifier to estimate divergence between distributions



Generative Adversarial Networks (GANs)

Data distribution: $p(x)$

Generator: $F_\theta : \mathcal{Z} \to \mathcal{X}$, $\quad q_\theta(x) : x = F_\theta(z), \quad z \sim q(z)$

Discriminator: $u_\phi : \mathcal{X} \to \mathbb{R}$, $\quad \widehat{P}_\phi(y = \text{data}|x) = \dfrac{1}{1 + \exp(-u_\phi(x))}$

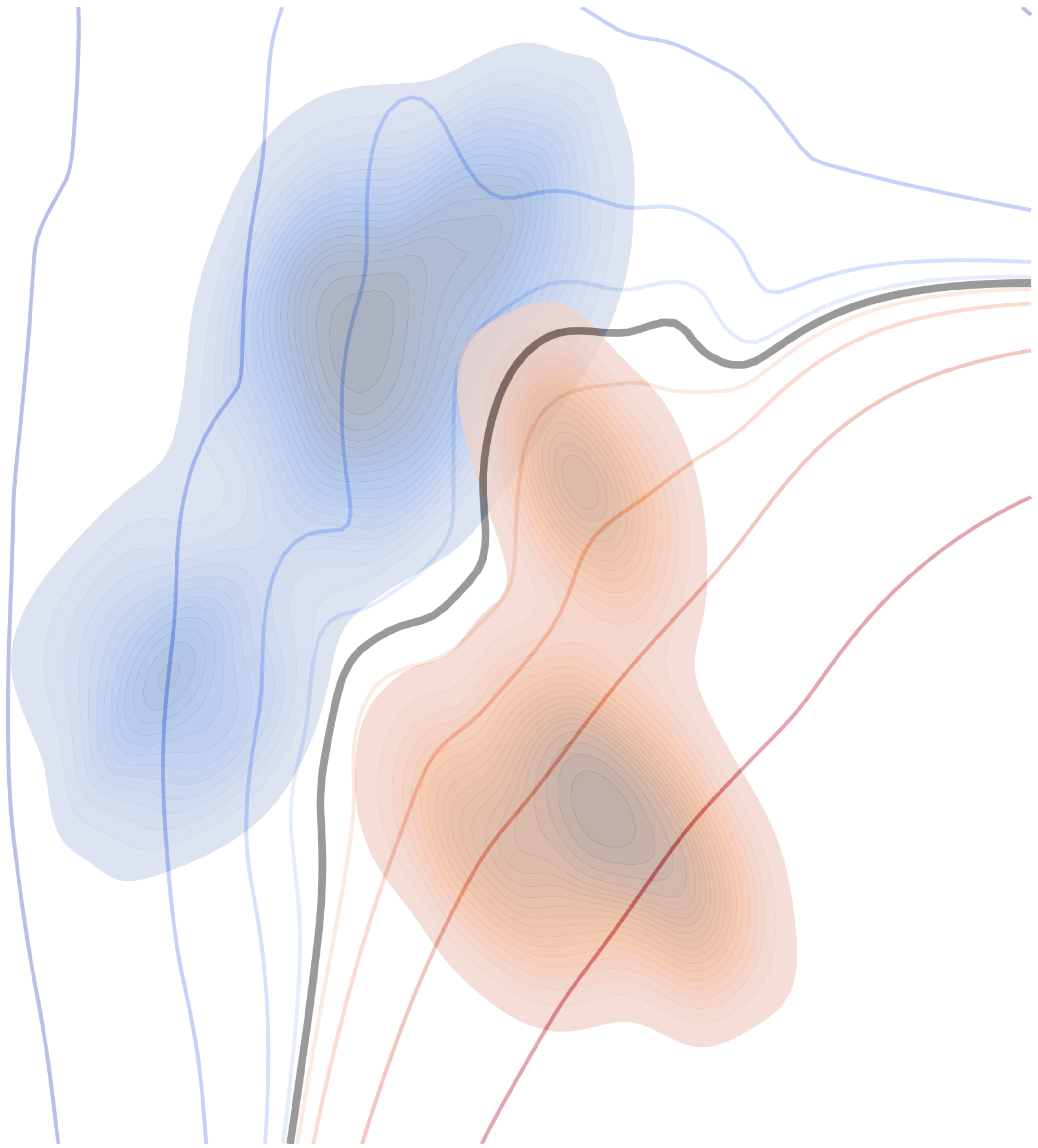
Guidance for Generative Model Training

$$\mathrm{JSD}(p\,\|\,q) = \frac{1}{2}\mathrm{KL}\left(p\,\middle\|\,\frac{p+q}{2}\right) + \frac{1}{2}\mathrm{KL}\left(q\,\middle\|\,\frac{p+q}{2}\right)$$

$$\mathrm{JSD}(p\,\|\,q) = \log(2) - \frac{1}{2}\min_{u:\mathcal{X}\to\mathbb{R}} \mathcal{L}(p,q,u)$$

$$\mathcal{L}(p,q,u) = \mathbb{E}_{p(x)}\big[\log(1+\exp(-u(x)))\big] + \mathbb{E}_{q(x)}\big[\log(1+\exp(u(x)))\big]$$

Idea: train a probabilistic classifier to estimate divergence between distributions



Generative Adversarial Networks (GANs)

Data distribution: $p(x)$

Generator: $F_\theta : \mathcal{Z} \to \mathcal{X}, \quad q_\theta(x) : x = F_\theta(z), \quad z \sim q(z)$

Discriminator: $u_\phi : \mathcal{X} \to \mathbb{R}, \quad \widehat{P}_\phi(y = \text{data}|x) = \dfrac{1}{1+\exp(-u_\phi(x))}$

$$u^* = \operatorname*{argmin}_{u:\mathcal{X}\to\mathbb{R}} \mathcal{L}(p,q,u), \quad u^*(x) = \log\frac{p(x)}{q(x)}, \quad P^*(y=\text{data}|x) = \frac{p(x)}{p(x)+q(x)}$$

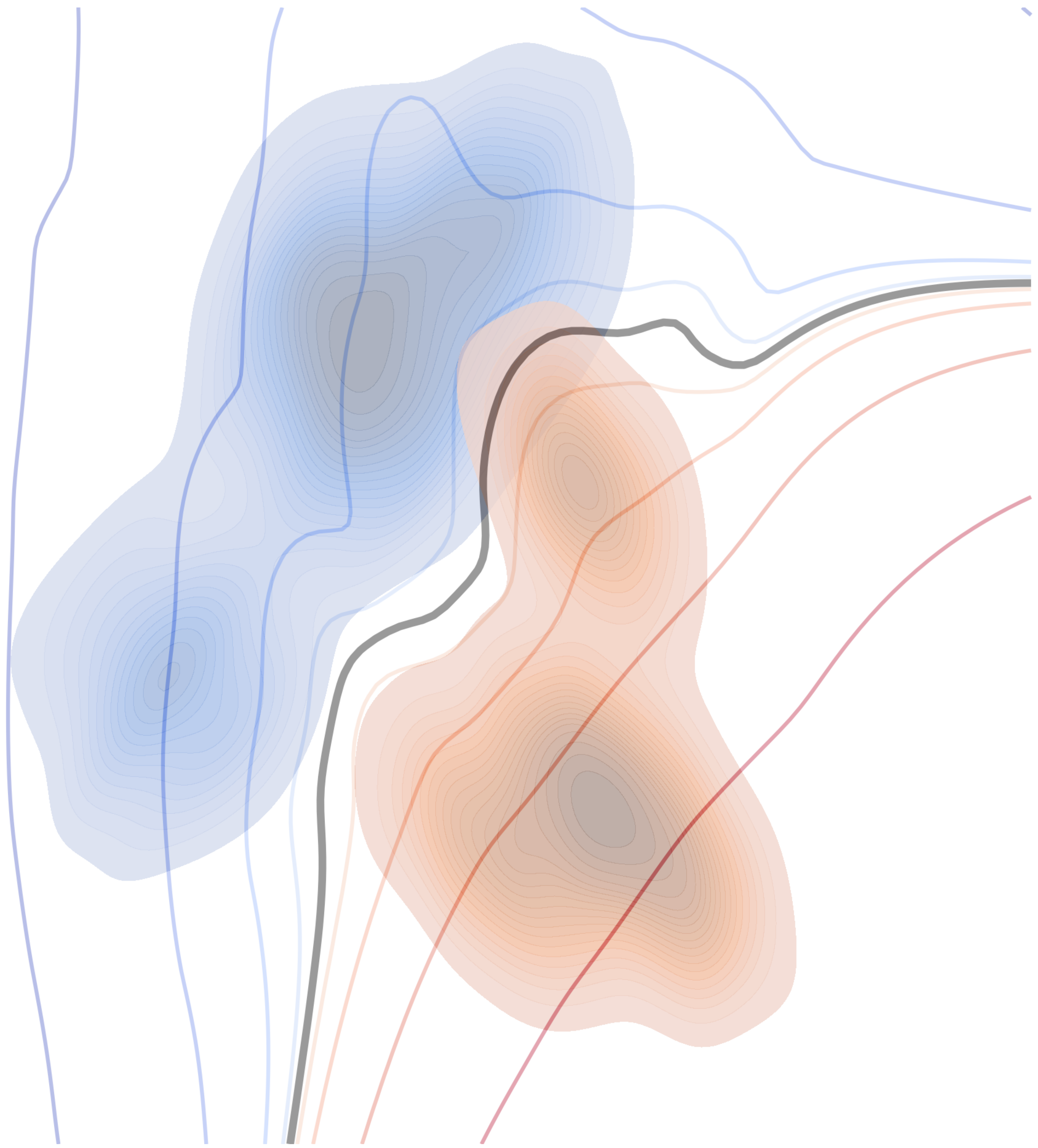
Guidance for Generative Model Training

$$\mathrm{JSD}(p \,\|\, q) = \frac{1}{2}\,\mathrm{KL}\left(p \,\middle\|\, \frac{p+q}{2}\right) + \frac{1}{2}\,\mathrm{KL}\left(q \,\middle\|\, \frac{p+q}{2}\right)$$

$$\mathrm{JSD}(p \,\|\, q) = \log(2) - \frac{1}{2}\min_{u:\mathcal{X}\to\mathbb{R}} \mathcal{L}(p, q, u)$$

$$\mathcal{L}(p, q, u) = \mathbb{E}_{p(x)}\big[\log(1 + \exp(-u(x)))\big] + \mathbb{E}_{q(x)}\big[\log(1 + \exp(u(x)))\big]$$

Idea: train a probabilistic classifier to estimate divergence between distributions



Generative Adversarial Networks (GANs)

Data distribution: $p(x)$

Generator: $F_\theta : \mathcal{Z} \to \mathcal{X}$, $\quad q_\theta(x) : x = F_\theta(z), \quad z \sim q(z)$

Discriminator: $u_\phi : \mathcal{X} \to \mathbb{R}$, $\quad \widehat{P}_\phi(y = \text{data}|x) = \dfrac{1}{1 + \exp(-u_\phi(x))}$

$$u^* = \operatorname*{argmin}_{u:\mathcal{X}\to\mathbb{R}} \mathcal{L}(p, q, u), \quad u^*(x) = \log\frac{p(x)}{q(x)}, \quad P^*(y = \text{data}|x) = \frac{p(x)}{p(x) + q(x)}$$

$$\mathcal{L}(p, q, u^*(p, q)) = -2\,\mathrm{JSD}(p \,\|\, q) + \log(4)$$

Max-Min game: $\max_{q_\theta} \min_{u_\phi} \mathcal{L}(p, q_\theta, u_\phi)$

Equilibrium: $q^* = p^*, u^* = 0$

Training Dynamics

Max-Min game: $\max_{q_\theta} \min_{u_\phi} \mathcal{L}(p, q_\theta, u_\phi)$

Equilibrium: $q^* = p^*, u^* = 0$

Training Dynamics

Max-Min game: $\max_{q_\theta} \min_{u_\phi} \mathcal{L}(p, q_\theta, u_\phi)$

Equilibrium: $q^* = p^*, u^* = 0$

In Theory:

Train **optimal discriminator** u^* by minimizing $\mathcal{L}(p, q_\theta, u)$

Update F_θ with gradient of $\mathcal{L}(p, q_\theta, u^*)$ given **optimal discriminator** u^*

Training Dynamics

Max-Min game: $\max_{q_\theta} \min_{u_\phi} \mathcal{L}(p, q_\theta, u_\phi)$

Equilibrium: $q^* = p^*, u^* = 0$

In Theory:

Train **optimal discriminator** u^* by minimizing $\mathcal{L}(p, q_\theta, u)$

Update F_θ with gradient of $\mathcal{L}(p, q_\theta, u^*)$ given **optimal discriminator** u^*

In Practice:

Update u_ϕ with **a few optimization steps** on $\mathcal{L}(p, q_\theta, u)$

Update F_θ with gradient of $\mathcal{L}(p, q_\theta, u_\phi)$ given **current discriminator** u_ϕ

Training Dynamics

Max-Min game: $\max_{q_\theta} \min_{u_\phi} \mathcal{L}(p, q_\theta, u_\phi)$

Equilibrium: $q^* = p^*, u^* = 0$

In Theory:

Train **optimal discriminator** u^* by minimizing $\mathcal{L}(p, q_\theta, u)$

Update F_θ with gradient of $\mathcal{L}(p, q_\theta, u^*)$ given **optimal discriminator** u^*

In Practice:

Update u_ϕ with **a few optimization steps** on $\mathcal{L}(p, q_\theta, u)$

Update F_θ with gradient of $\mathcal{L}(p, q_\theta, u_\phi)$ given **current discriminator** u_ϕ

- The generator receives training signal through the discriminator
- Even if the generator is perfectly aligned with the target, a suboptimal discriminator can destroy the alignment
- The instability of alignment complicates training dynamics
- Training can get into a limit cycle and never reach the equilibrium

Training Dynamics

Max-Min game: $\max_{q_\theta} \min_{u_\phi} \mathcal{L}(p, q_\theta, u_\phi)$

Equilibrium: $q^* = p^*, u^* = 0$

In Theory:

Train **optimal discriminator** u^* by minimizing $\mathcal{L}(p, q_\theta, u)$

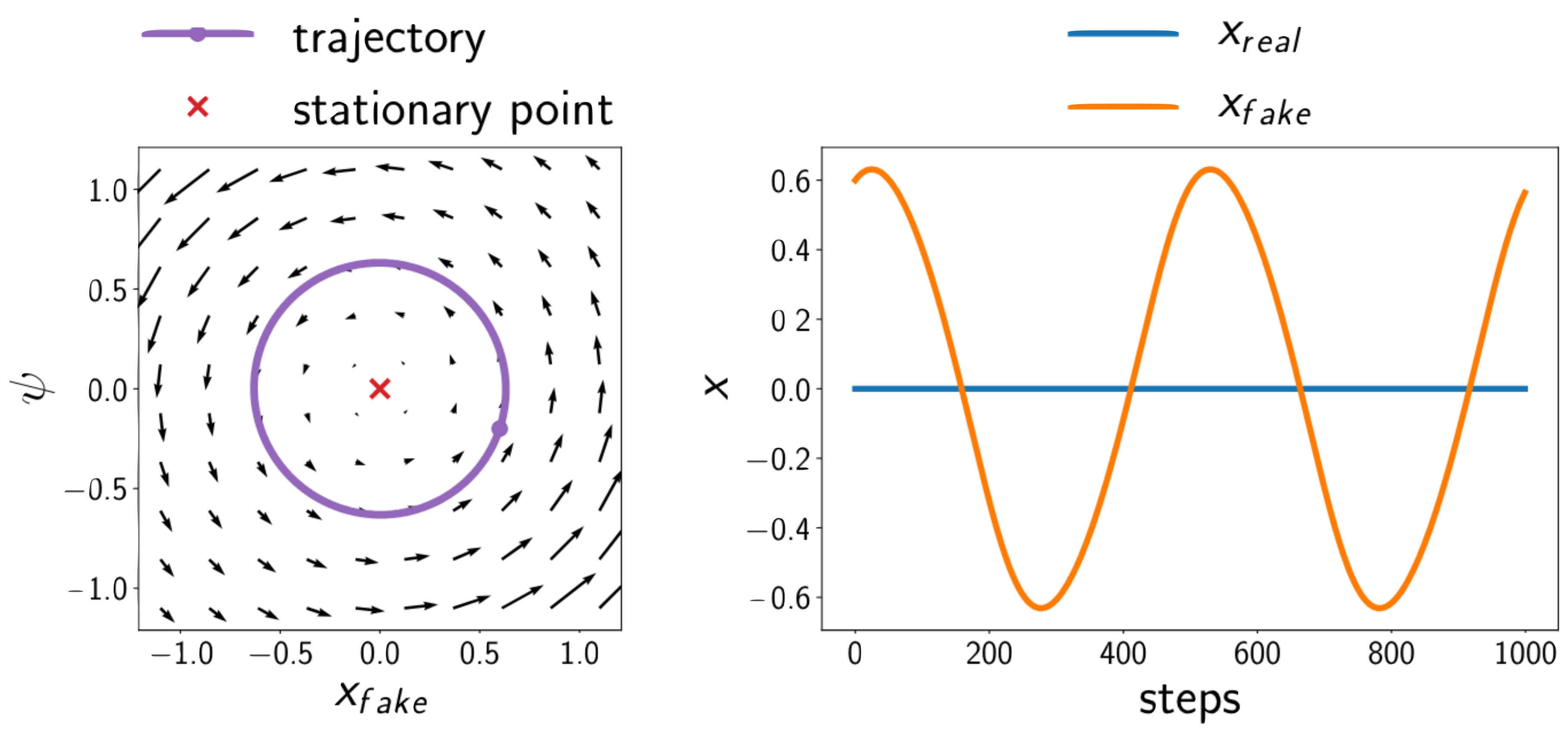
Update F_θ with gradient of $\mathcal{L}(p, q_\theta, u^*)$ given **optimal discriminator** u^*

In Practice:

Update u_ϕ with **a few optimization steps** on $\mathcal{L}(p, q_\theta, u)$

Update F_θ with gradient of $\mathcal{L}(p, q_\theta, u_\phi)$ given **current discriminator** u_ϕ

- The generator receives training signal through the discriminator
- Even if the generator is perfectly aligned with the target, a suboptimal discriminator can destroy the alignment
- The instability of alignment complicates training dynamics
- Training can get into a limit cycle and never reach the equilibrium



Training Dynamics

Max-Min game: $\max_{q_\theta} \min_{u_\phi} \mathcal{L}(p, q_\theta, u_\phi)$

Equilibrium: $q^* = p^*, u^* = 0$

In Theory:

Train **optimal discriminator** u^* by minimizing $\mathcal{L}(p, q_\theta, u)$

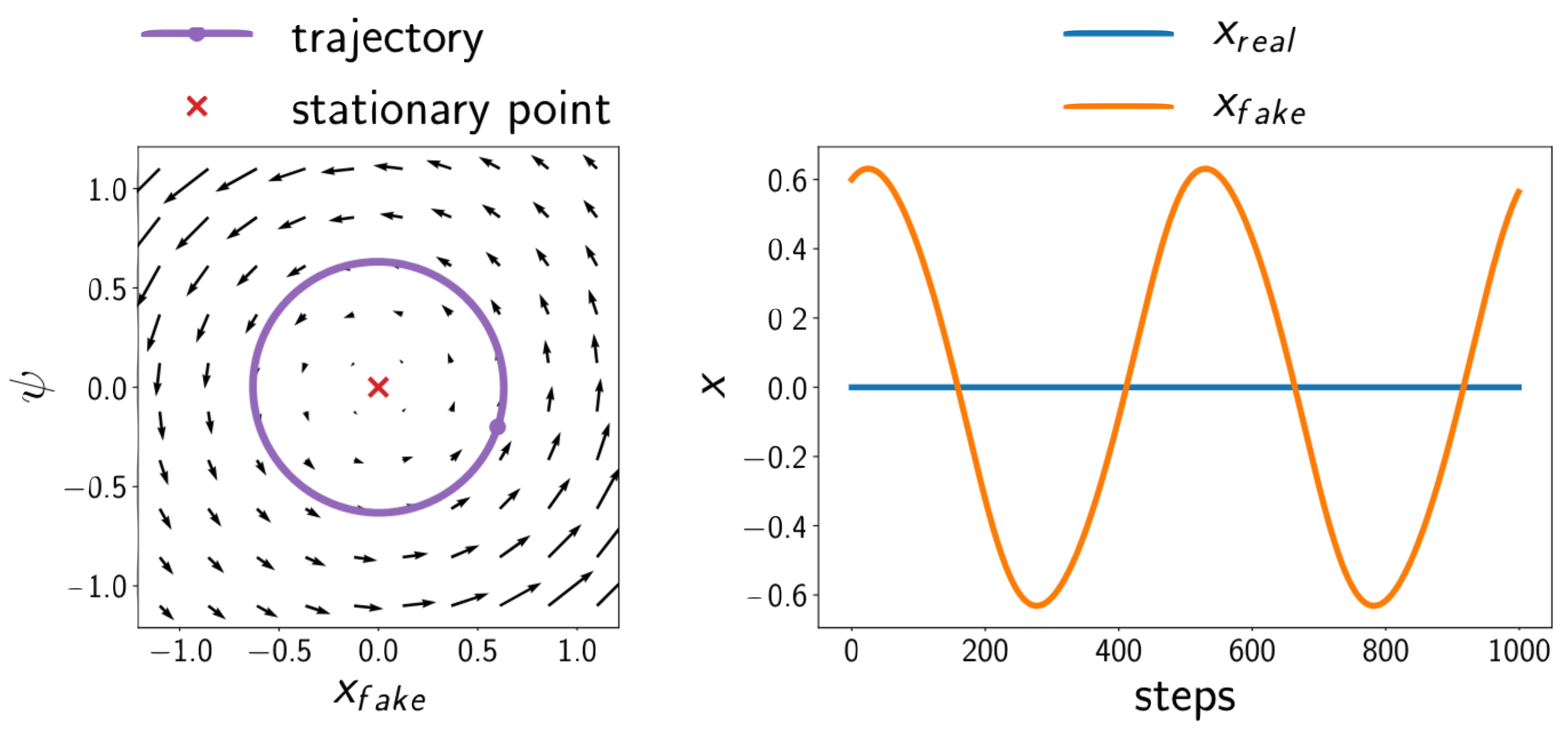
Update F_θ with gradient of $\mathcal{L}(p, q_\theta, u^*)$ given **optimal discriminator** u^*

In Practice:

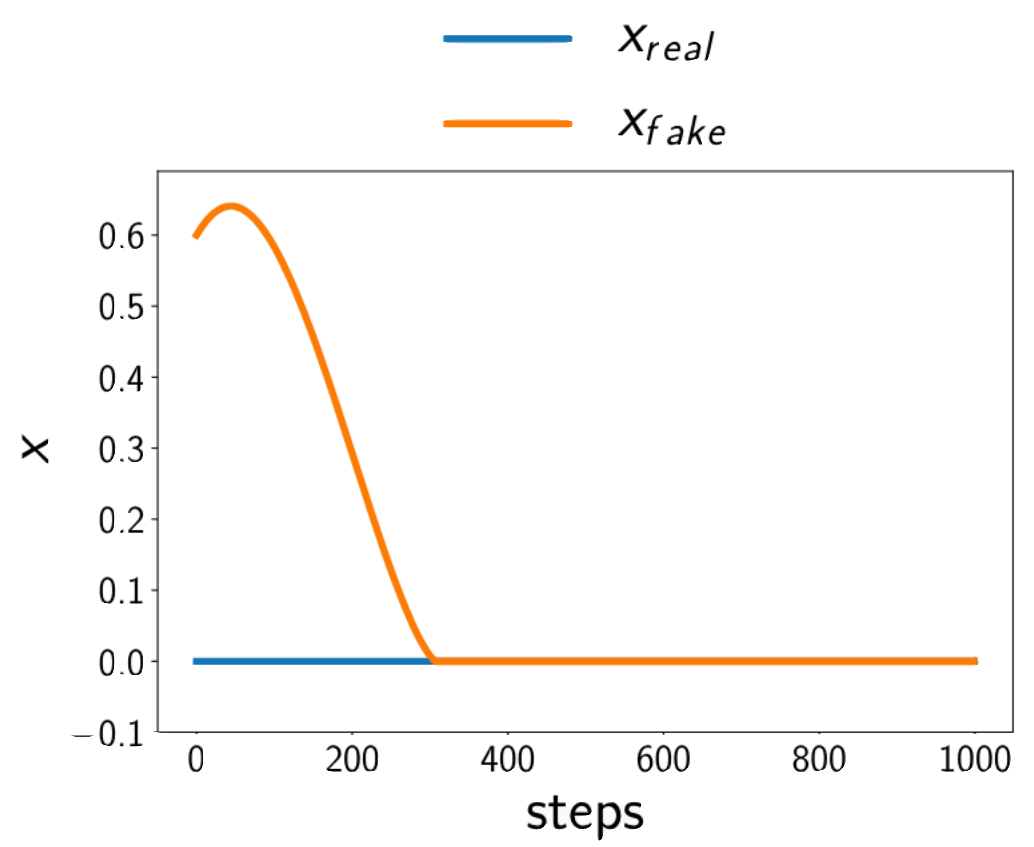
Update u_ϕ with **a few optimization steps** on $\mathcal{L}(p, q_\theta, u)$

Update F_θ with gradient of $\mathcal{L}(p, q_\theta, u_\phi)$ given **current discriminator** u_ϕ

- The generator receives training signal through the discriminator
- Even if the generator is perfectly aligned with the target, a suboptimal discriminator can destroy the alignment
- The instability of alignment complicates training dynamics
- Training can get into a limit cycle and never reach the equilibrium



Can we design training objectives that preserve distribution alignment?



How to Preserve the Alignment?

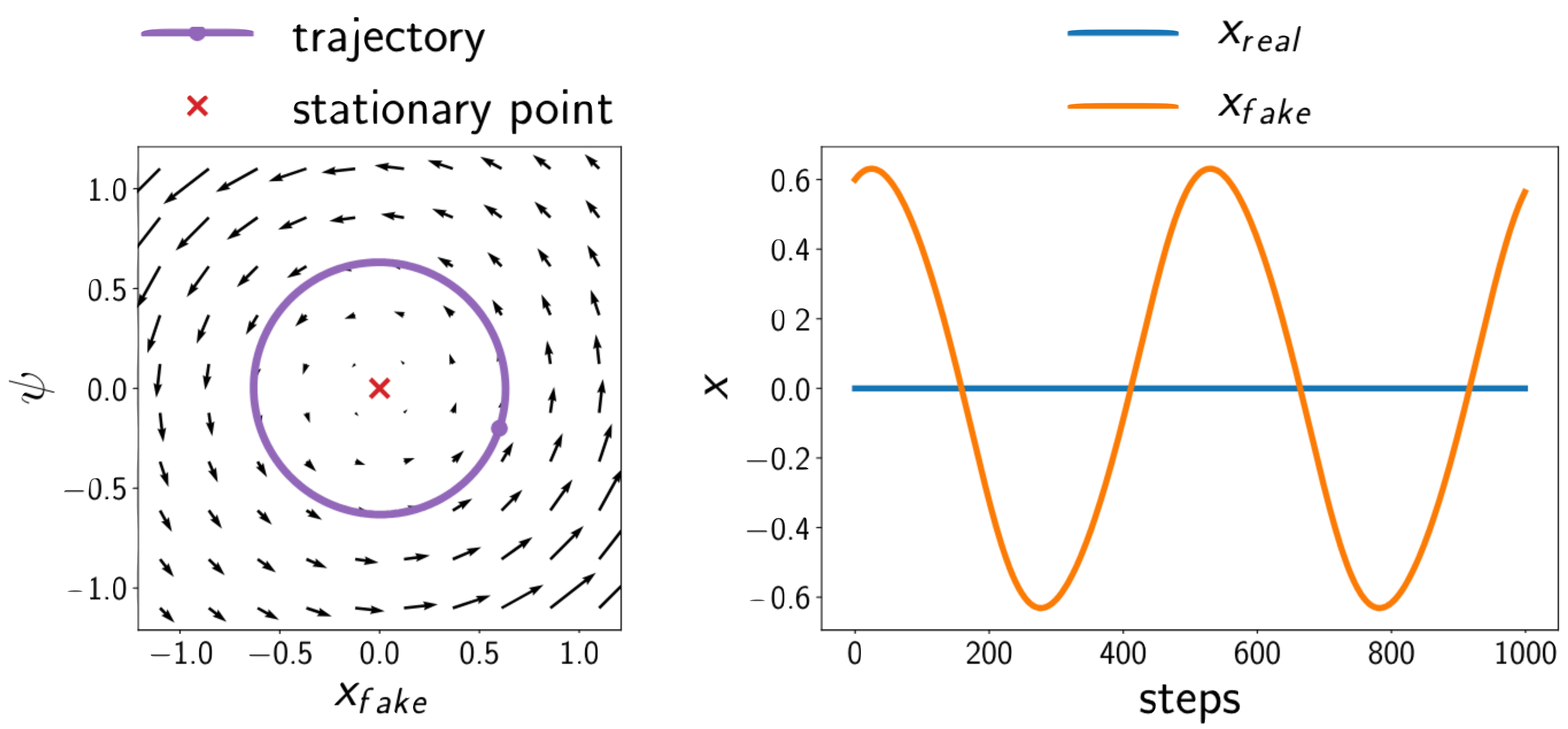
Training with **unary discriminator** $u(\cdot)$

$u(\cdot)$ distinguishes between **real samples** $x \sim p(x)$

and **generated samples** $x \sim q(x)$

$\mathcal{L}_G(q, u) = \mathbb{E}_{q(x)}\big[S(u(x))\big] = \langle a_u^S, q \rangle$

$\nabla_q \mathcal{L}_G(q, u) = a_u^S$



How to Preserve the Alignment?

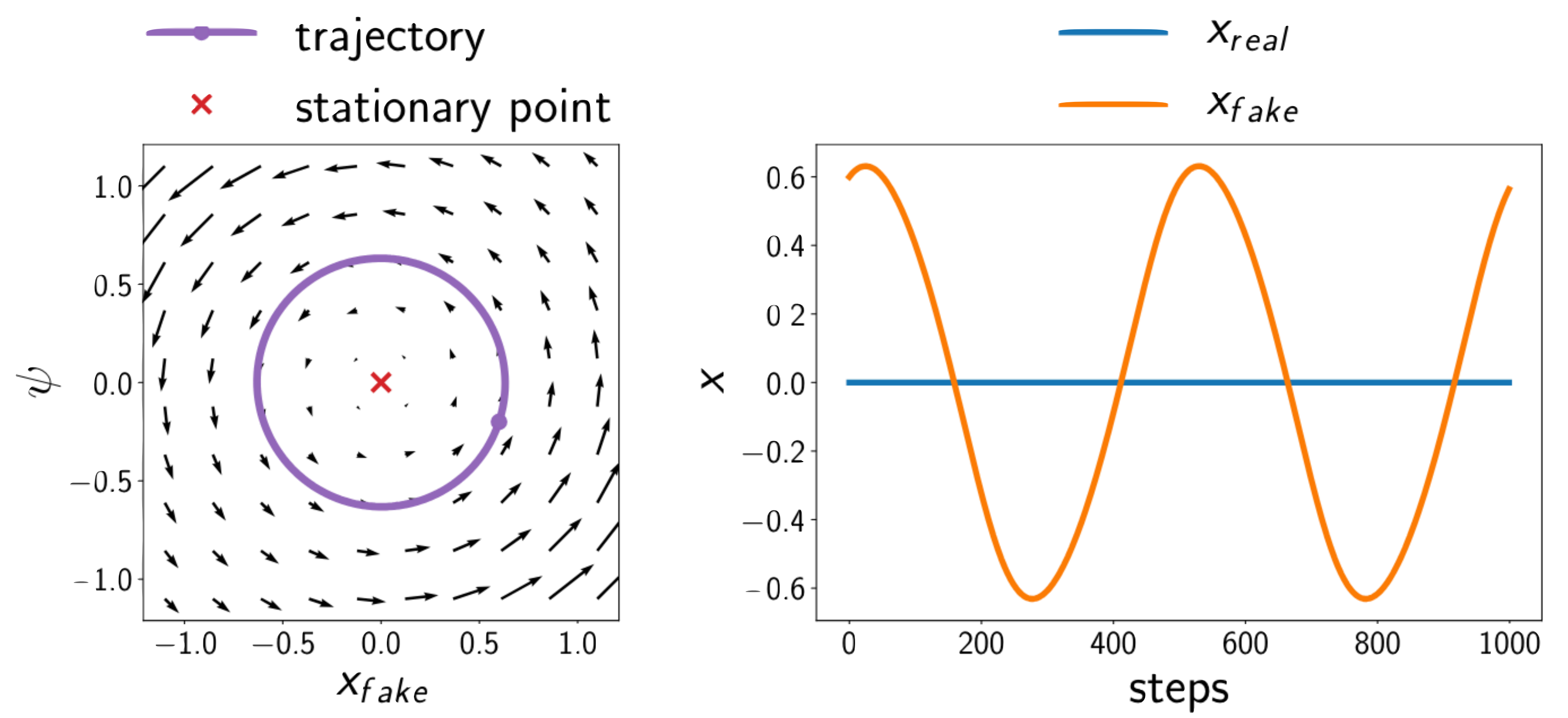
Training with **unary discriminator** $u(\cdot)$

$u(\cdot)$ distinguishes between **real samples** $x \sim p(x)$

and **generated samples** $x \sim q(x)$

$$\mathcal{L}_G(q,u) = \mathbb{E}_{q(x)}\big[S(u(x))\big] = \langle a_u^S, q\rangle$$

$$\nabla_q \mathcal{L}_G(q,u) = a_u^S$$



- The gradient depends only on the discriminator
- The alignment is preserved only if the discriminator is optimal

How to Preserve the Alignment?

Training with **unary discriminator** $u(\cdot)$

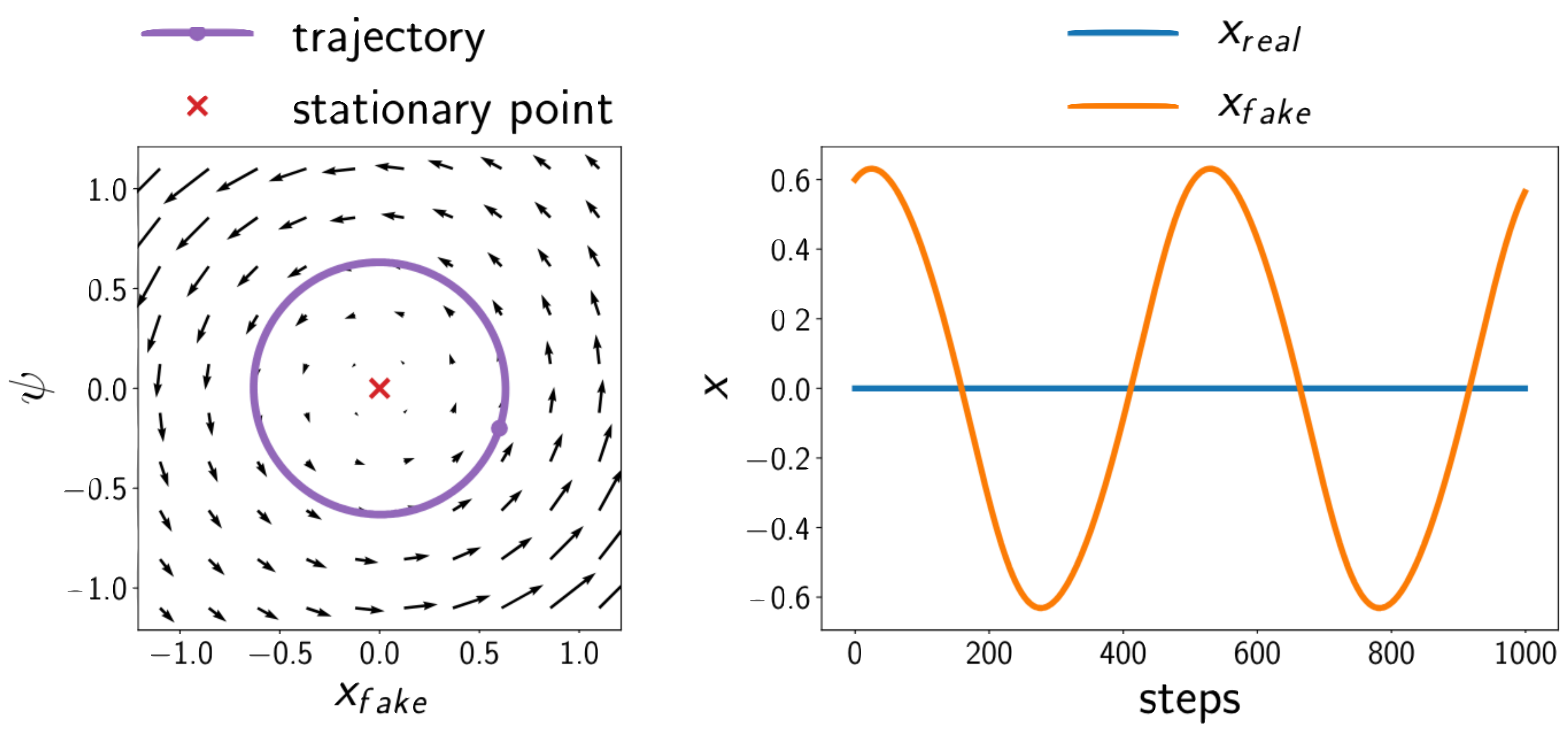
$u(\cdot)$ distinguishes between **real samples** $x \sim p(x)$

and **generated samples** $x \sim q(x)$

$$\mathcal{L}_G(q,u) = \mathbb{E}_{q(x)}\big[S(u(x))\big] = \langle a_u^S, q\rangle$$

$$\nabla_q \mathcal{L}_G(q,u) = a_u^S$$

- The gradient depends only on the discriminator
- The alignment is preserved only if the discriminator is optimal



Training with **symmetric binary discriminator** $U(\cdot,\cdot)$

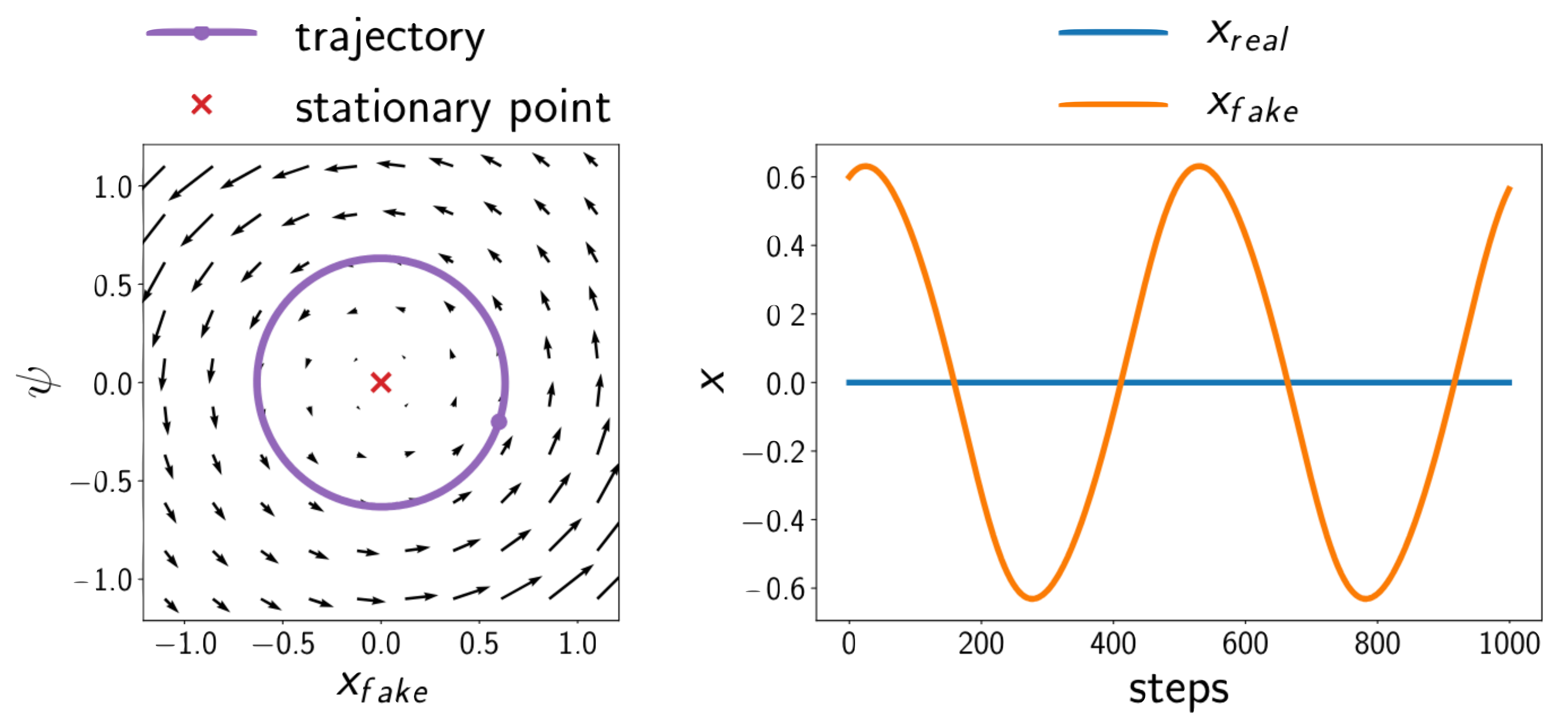
$U(\cdot,\cdot)$ distinguishes between **same-distribution pairs** $(x,y) \sim p(x)p(y), q(x)q(y)$

and **different-distribution pairs** $(x,y) \sim p(x)q(y), p(y)q(x)$

$$\mathcal{L}_G(q,U) = \mathbb{E}_{p\times p}[S(U(x,y))] + \mathbb{E}_{q\times q}[S(U(x,y))] - 2\mathbb{E}_{p\times q}[S(U(x,y))]$$
$$= \langle q-p, A_U^S(q-p)\rangle$$

$$\nabla_q \mathcal{L}_G(q,U) = 2A_U^S(q-p)$$

How to Preserve the Alignment?



Training with **unary discriminator** $u(\cdot)$

$u(\cdot)$ distinguishes between **real samples** $x \sim p(x)$

and **generated samples** $x \sim q(x)$

$$\mathcal{L}_G(q, u) = \mathbb{E}_{q(x)}\big[S(u(x))\big] = \langle a_u^S, q \rangle$$

$$\nabla_q \mathcal{L}_G(q, u) = a_u^S$$

- The gradient depends only on the discriminator
- The alignment is preserved only if the discriminator is optimal

Training with **symmetric binary discriminator** $U(\cdot, \cdot)$

$U(\cdot, \cdot)$ distinguishes between **same-distribution pairs** $(x, y) \sim p(x)p(y), q(x)q(y)$

and **different-distribution pairs** $(x, y) \sim p(x)q(y), p(y)q(x)$

$$\mathcal{L}_G(q, U) = \mathbb{E}_{p\times p}[S(U(x, y))] + \mathbb{E}_{q\times q}[S(U(x, y))] - 2\mathbb{E}_{p\times q}[S(U(x, y))]$$
$$= \langle q - p, A_U^S(q - p) \rangle$$

$$\nabla_q \mathcal{L}_G(q, U) = 2A_U^S(q - p)$$

- The gradient is a function of the discriminator and the deviation from the target
- The alignment, once achieved, is preserved with any discriminator

How to Preserve the Alignment?

Training with **unary discriminator** $u(\cdot)$

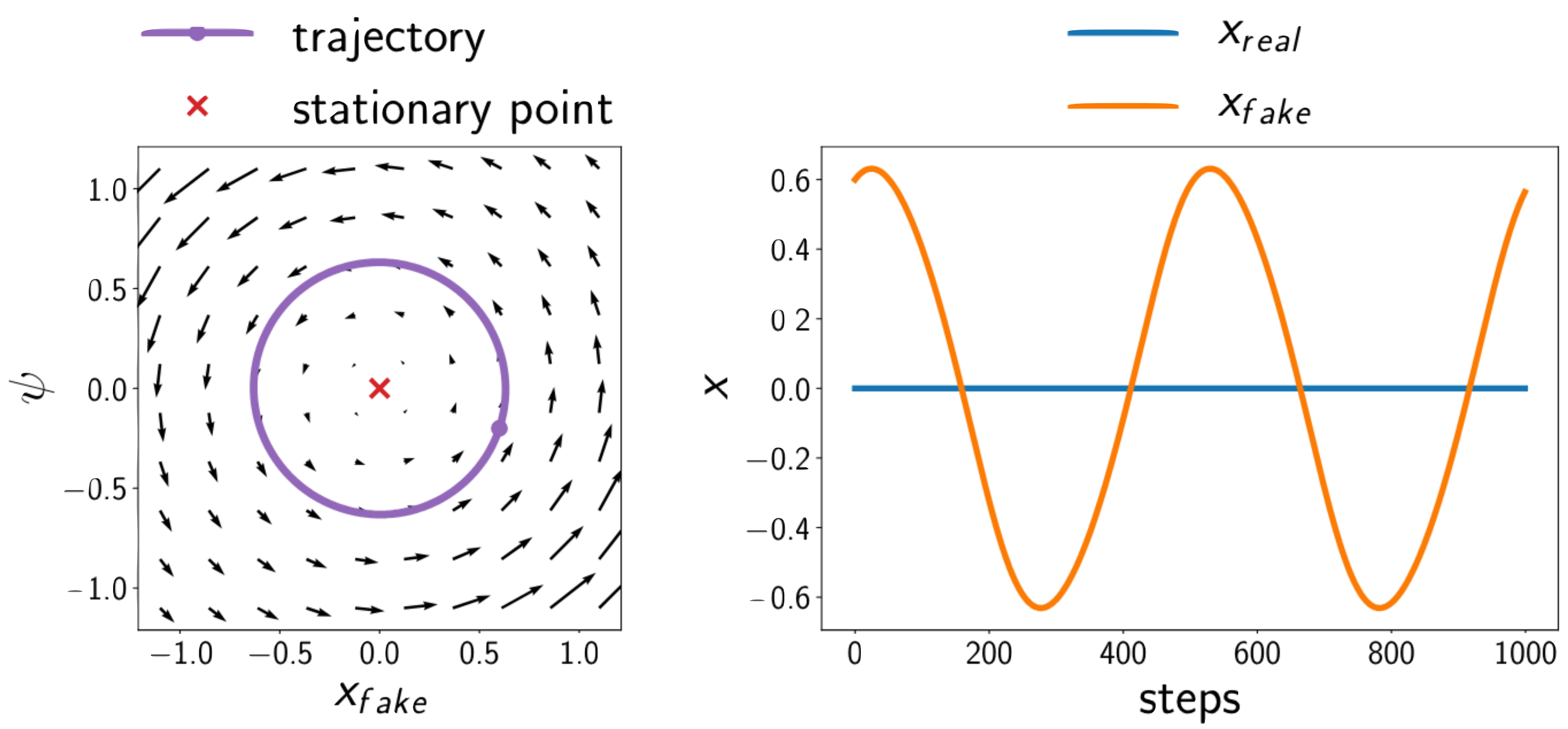
$u(\cdot)$ distinguishes between **real samples** $x \sim p(x)$

and **generated samples** $x \sim q(x)$

$$\mathcal{L}_G(q, u) = \mathbb{E}_{q(x)}\left[S(u(x))\right] = \langle a_u^S, q \rangle$$

$$\nabla_q \mathcal{L}_G(q, u) = a_u^S$$

- The gradient depends only on the discriminator
- The alignment is preserved only if the discriminator is optimal



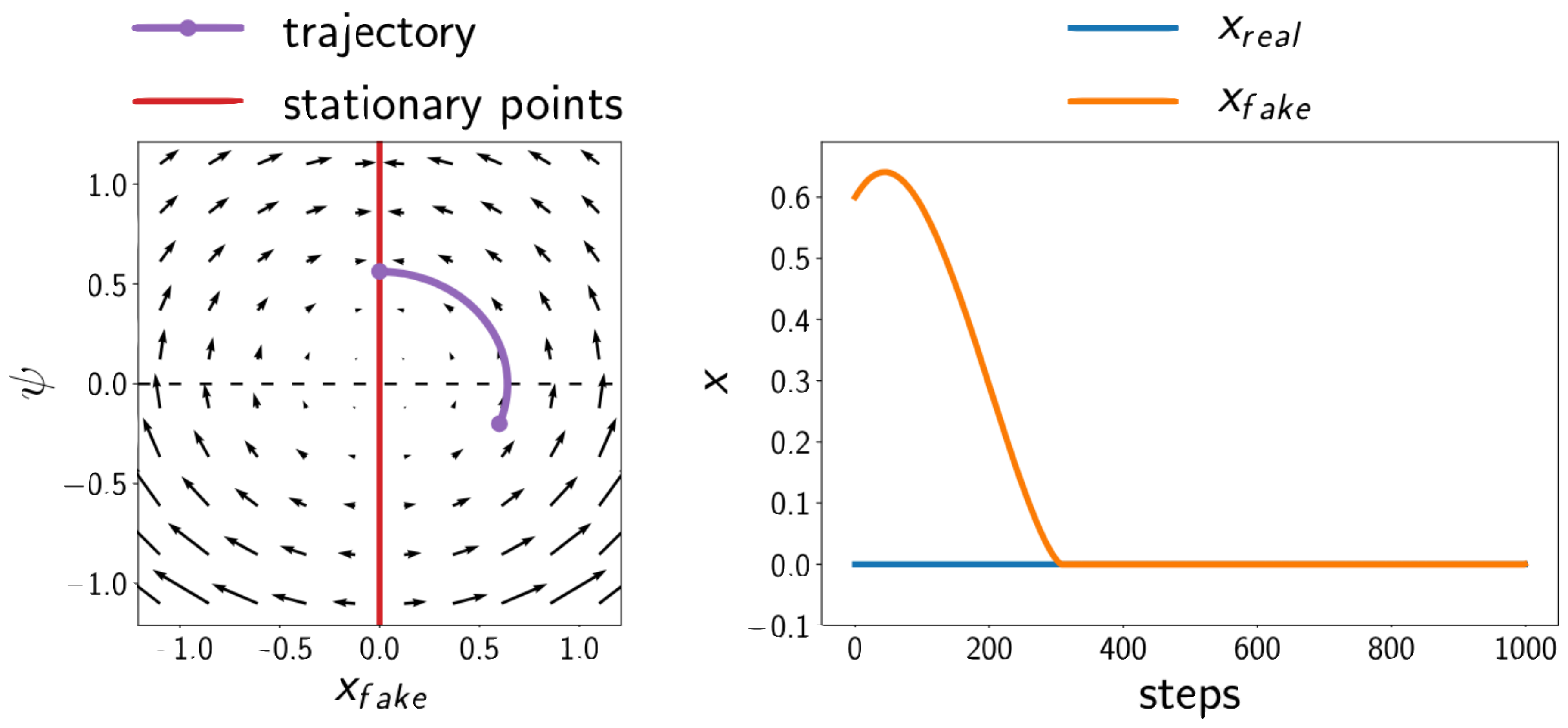
Training with **symmetric binary discriminator** $U(\cdot, \cdot)$

$U(\cdot, \cdot)$ distinguishes between **same-distribution pairs** $(x, y) \sim p(x)p(y), q(x)q(y)$

and **different-distribution pairs** $(x, y) \sim p(x)q(y), p(y)q(x)$

$$\mathcal{L}_G(q, U) = \mathbb{E}_{p\times p}[S(U(x, y))] + \mathbb{E}_{q\times q}[S(U(x, y))] - 2\mathbb{E}_{p\times q}[S(U(x, y))]$$
$$= \langle q - p, A_U^S(q - p) \rangle$$

$$\nabla_q \mathcal{L}_G(q, U) = 2A_U^S(q - p)$$



- The gradient is a function of the discriminator and the deviation from the target
- The alignment, once achieved, is preserved with any discriminator

PairGAN

A quadratic form in the space of distributions

$$\mathcal{L}_G(q, U) = \langle q - p, A_U^S (q - p) \rangle$$

corresponds to the Maximum Mean Discrepancy (MMD) distance

$$\mathrm{MMD}^2(p, q) = \langle q - p, A(q - p) \rangle = \|p - q\|_A^2$$

if A is a positive-definite kernel

PairGAN

A quadratic form in the space of distributions

$$\mathcal{L}_G(q, U) = \langle q - p, A_U^S(q - p) \rangle$$

corresponds to the Maximum Mean Discrepancy (MMD) distance

$$\mathrm{MMD}^2(p, q) = \langle q - p, A(q - p) \rangle = \|p - q\|_A^2$$

if A is a positive-definite kernel

In high-dimensional spaces, manually-designed kernels provide weak signals

PairGAN

A quadratic form in the space of distributions

$$\mathcal{L}_G(q, U) = \langle q - p, A_U^S(q - p) \rangle$$

corresponds to the Maximum Mean Discrepancy (MMD) distance

$$\mathrm{MMD}^2(p, q) = \langle q - p, A(q - p) \rangle = \|p - q\|_A^2$$

if A is a positive-definite kernel

In high-dimensional spaces, manually-designed kernels provide weak signals

Instead, an adaptive pairwise discriminator can be trained with the log-likelihood objective

$$\begin{aligned}\mathcal{L}_D(q, U) = \mathbb{E}_{p \times p}\big[-\log \sigma(U(x, y)) \big] + \mathbb{E}_{q \times q}\big[-\log \sigma(U(x, y)) \big] \\ + \mathbb{E}_{p \times q}\big[-\log \sigma(-U(x, y)) \big] + \mathbb{E}_{q \times p}\big[-\log \sigma(-U(x, y)) \big]\end{aligned}$$

$$\begin{aligned}\mathcal{L}_G(q, U) = \mathbb{E}_{p \times p}\big[\log \sigma(U(x, y)) \big] + \mathbb{E}_{q \times q}\big[\log \sigma(U(x, y)) \big] \\ - 2\mathbb{E}_{p \times q}\big[\log \sigma(U(x, y)) \big]\end{aligned}$$

PairGAN

A quadratic form in the space of distributions

$$\mathcal{L}_G(q, U) = \langle q - p, A_U^S(q - p)\rangle$$

corresponds to the Maximum Mean Discrepancy (MMD) distance

$$\mathrm{MMD}^2(p, q) = \langle q - p, A(q - p)\rangle = \|p - q\|_A^2$$

if A is a positive-definite kernel

In high-dimensional spaces, manually-designed kernels provide weak signals

Instead, an adaptive pairwise discriminator can be trained with the log-likelihood objective

$$\begin{aligned}\mathcal{L}_D(q, U) = {} & \mathbb{E}_{p\times p}\big[-\log\sigma(U(x,y))\big] + \mathbb{E}_{q\times q}\big[-\log\sigma(U(x,y))\big] \\ & + \mathbb{E}_{p\times q}\big[-\log\sigma(-U(x,y))\big] + \mathbb{E}_{q\times p}\big[-\log\sigma(-U(x,y))\big]\end{aligned}$$

$$\begin{aligned}\mathcal{L}_G(q, U) = {} & \mathbb{E}_{p\times p}\big[\log\sigma(U(x,y))\big] + \mathbb{E}_{q\times q}\big[\log\sigma(U(x,y))\big] \\ & - 2\mathbb{E}_{p\times q}\big[\log\sigma(U(x,y))\big]\end{aligned}$$

Non-zero-sum PairGAN

$$\min_{U_\phi} \ \mathcal{L}_D(q_\theta, U_\phi)$$

$$\min_{q_\theta} \ \mathcal{L}_G(q_\theta, U_\phi)$$

PairGAN

A quadratic form in the space of distributions

$$\mathcal{L}_G(q, U) = \langle q - p, A_U^S(q - p) \rangle$$

corresponds to the Maximum Mean Discrepancy (MMD) distance

$$\mathrm{MMD}^2(p, q) = \langle q - p, A(q - p) \rangle = \|p - q\|_A^2$$

if A is a positive-definite kernel

In high-dimensional spaces, manually-designed kernels provide weak signals

Instead, an adaptive pairwise discriminator can be trained with the log-likelihood objective

$$\begin{aligned}\mathcal{L}_D(q, U) = &\ \mathbb{E}_{p\times p}\big[-\log\sigma(U(x,y))\big] + \mathbb{E}_{q\times q}\big[-\log\sigma(U(x,y))\big] \\ &+ \mathbb{E}_{p\times q}\big[-\log\sigma(-U(x,y))\big] + \mathbb{E}_{q\times p}\big[-\log\sigma(-U(x,y))\big]\end{aligned}$$

$$\begin{aligned}\mathcal{L}_G(q, U) = &\ \mathbb{E}_{p\times p}\big[\log\sigma(U(x,y))\big] + \mathbb{E}_{q\times q}\big[\log\sigma(U(x,y))\big] \\ &- 2\mathbb{E}_{p\times q}\big[\log\sigma(U(x,y))\big]\end{aligned}$$

Non-zero-sum PairGAN

$$\min_{U_\phi} \ \mathcal{L}_D(q_\theta, U_\phi)$$

$$\min_{q_\theta} \ \mathcal{L}_G(q_\theta, U_\phi)$$

Zero-sum PairGAN

$$\max_{U_\phi} \ \mathcal{L}_G(q_\theta, U_\phi)$$

$$\min_{q_\theta} \ \mathcal{L}_G(q_\theta, U_\phi)$$

Divergence Minimization

$$\mathcal{L}_D(q,U) = \mathbb{E}_{p\times p}\big[-\log\sigma(U(x,y))\big] + \mathbb{E}_{q\times q}\big[-\log\sigma(U(x,y))\big] + \mathbb{E}_{p\times q}\big[-\log\sigma(-U(x,y))\big] + \mathbb{E}_{q\times p}\big[-\log\sigma(-U(x,y))\big]$$

$$\mathcal{L}_G(q,U) = \mathbb{E}_{p\times p}\big[\log\sigma(U(x,y))\big] + \mathbb{E}_{q\times q}\big[\log\sigma(U(x,y))\big] - 2\mathbb{E}_{p\times q}\big[\log\sigma(U(x,y))\big]$$

Non-zero-sum PairGAN

$$\min_{U_\phi}\ \mathcal{L}_D(q_\theta, U_\phi)$$

$$\min_{q_\theta}\ \mathcal{L}_G(q_\theta, U_\phi)$$

Zero-sum PairGAN

$$\max_{U_\phi}\ \mathcal{L}_G(q_\theta, U_\phi)$$

$$\min_{q_\theta}\ \mathcal{L}_G(q_\theta, U_\phi)$$

Divergence Minimization

$$\mathcal{L}_D(q,U) = \mathbb{E}_{p\times p}\big[-\log\sigma(U(x,y))\big] + \mathbb{E}_{q\times q}\big[-\log\sigma(U(x,y))\big] + \mathbb{E}_{p\times q}\big[-\log\sigma(-U(x,y))\big] + \mathbb{E}_{q\times p}\big[-\log\sigma(-U(x,y))\big]$$

$$\mathcal{L}_G(q,U) = \mathbb{E}_{p\times p}\big[\log\sigma(U(x,y))\big] + \mathbb{E}_{q\times q}\big[\log\sigma(U(x,y))\big] - 2\mathbb{E}_{p\times q}\big[\log\sigma(U(x,y))\big]$$

Non-zero-sum PairGAN	**Zero-sum PairGAN**
$\min_{U_\phi} \mathcal{L}_D(q_\theta, U_\phi)$	$\max_{U_\phi} \mathcal{L}_G(q_\theta, U_\phi)$
$\min_{q_\theta} \mathcal{L}_G(q_\theta, U_\phi)$	$\min_{q_\theta} \mathcal{L}_G(q_\theta, U_\phi)$

Proposition (PairGAN Distribution Divergences).
Consider distributions $p(x)$ and $q(x)$. Let $M^+_{p,q}(x,y)$, $M^-_{p,q}(x,y)$, and $M_{p,q}(x,y)$ denote the mixture distributions over pairs

$$M^+_{p,q}(x,y) = \frac{1}{2}p(x)p(y) + \frac{1}{2}q(x)q(y)$$

$$M^-_{p,q}(x,y) = \frac{1}{2}p(x)q(y) + \frac{1}{2}q(x)p(y)$$

$$M_{p,q}(x,y) = \frac{1}{2}M^+_{p,q}(x,y) + \frac{1}{2}M^-_{p,q}(x,y)$$

Given optimal PairGAN discriminators (zero-sum or non-zero-sum), the PairGAN generator objectives are equivalent to the following distribution divergences

$$U^*_N(q) = \operatorname*{argmin}_{U:\mathcal{X}\to\mathbb{R}} \mathcal{L}_D(q,U)$$
$$\mathcal{L}_G(q,U^*_N(q)) = 4\cdot\big(\mathrm{KL}(M^+_{p,q}\,\|\,M_{p,q}) + \mathrm{KL}(M_{p,q}\,\|\,M^+_{p,q})\big)$$

$$U^*_Z(q) = \operatorname*{argmax}_{U:\mathcal{X}\to[\log(\varepsilon),\infty)} \mathcal{L}_G(q,U), \quad (0<\varepsilon<1)$$
$$\mathcal{L}_G(q,U^*_Z(q)) = -\log(\varepsilon)\cdot\mathrm{TV}(M^+_{p,q}\,\|\,M^-_{p,q}).$$

Consequently,

$\mathcal{L}_G(q,U^*(q)) \geq 0$ | $\mathcal{L}_G(q,U^*(q)) = 0$ *if and only if* $q=p$

Experimental validation

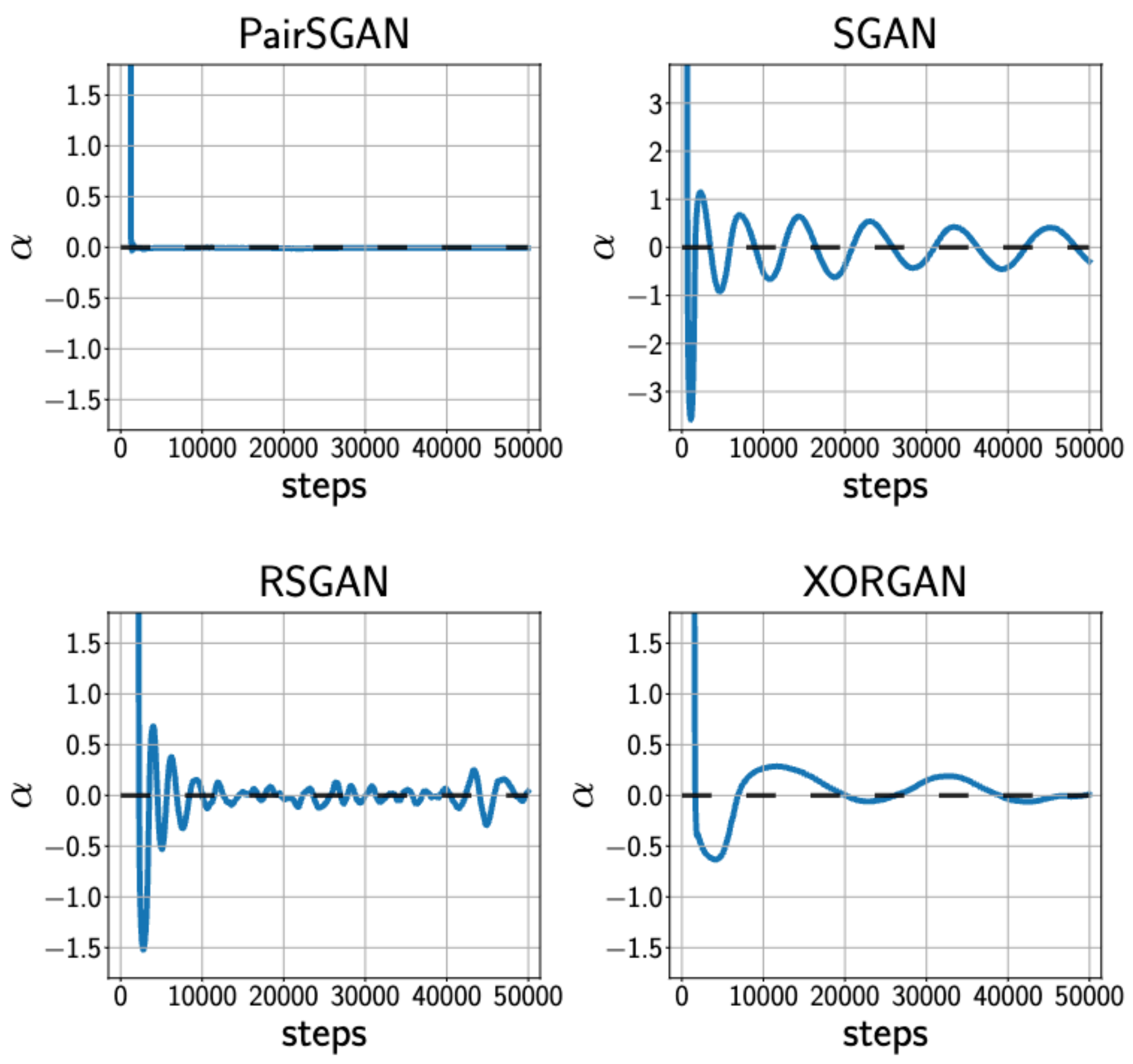
Experimental validation

- Verified alignment preservation in DCGAN with restricted generator

Experimental validation

- Verified alignment preservation in DCGAN with restricted generator

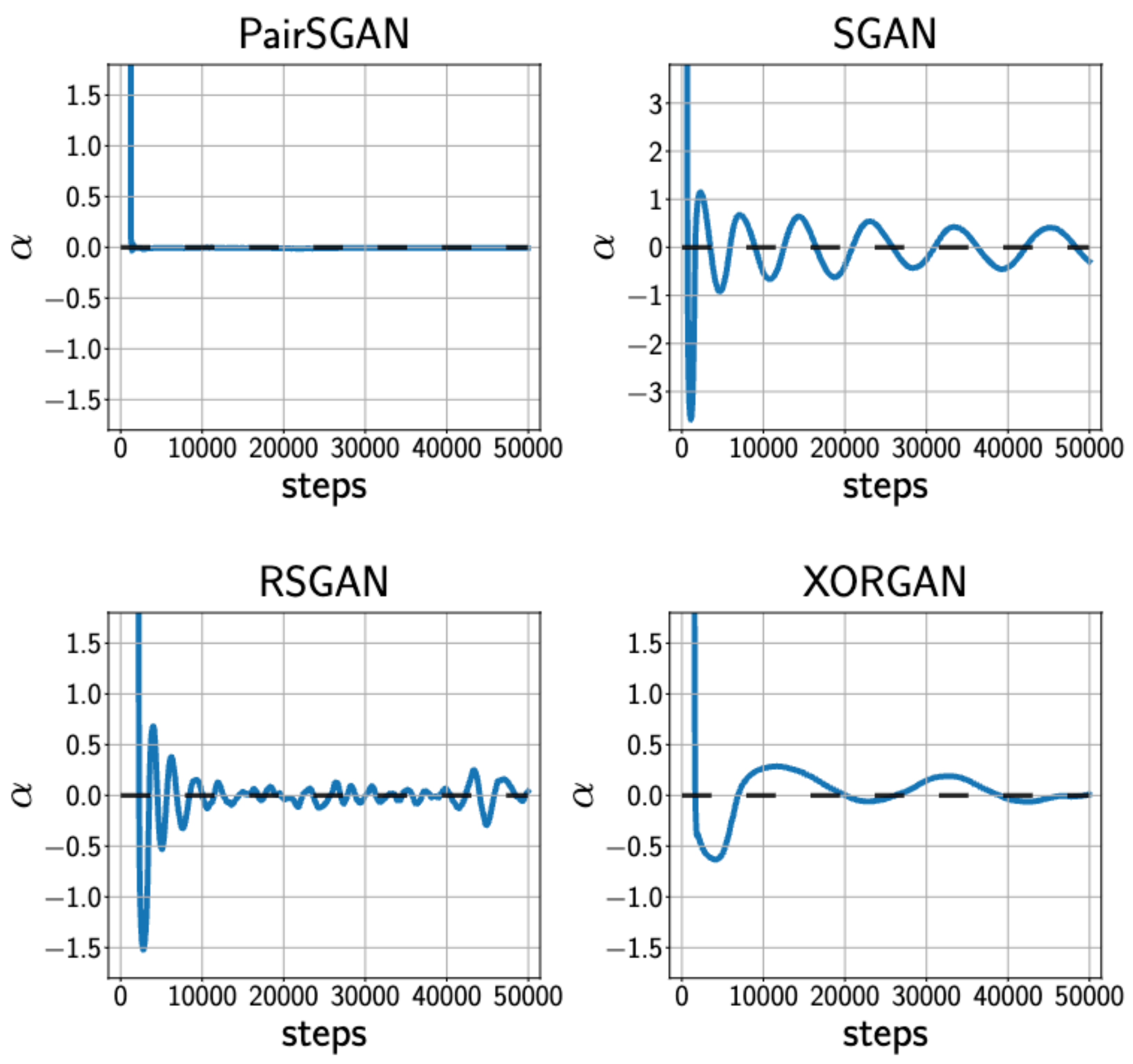
Alignment stability in
restricted DCGAN generator experiment



Experimental validation

- Verified alignment preservation in DCGAN with restricted generator
- Improved FID curve stability on CIFAR-10 and CAT benchmarks

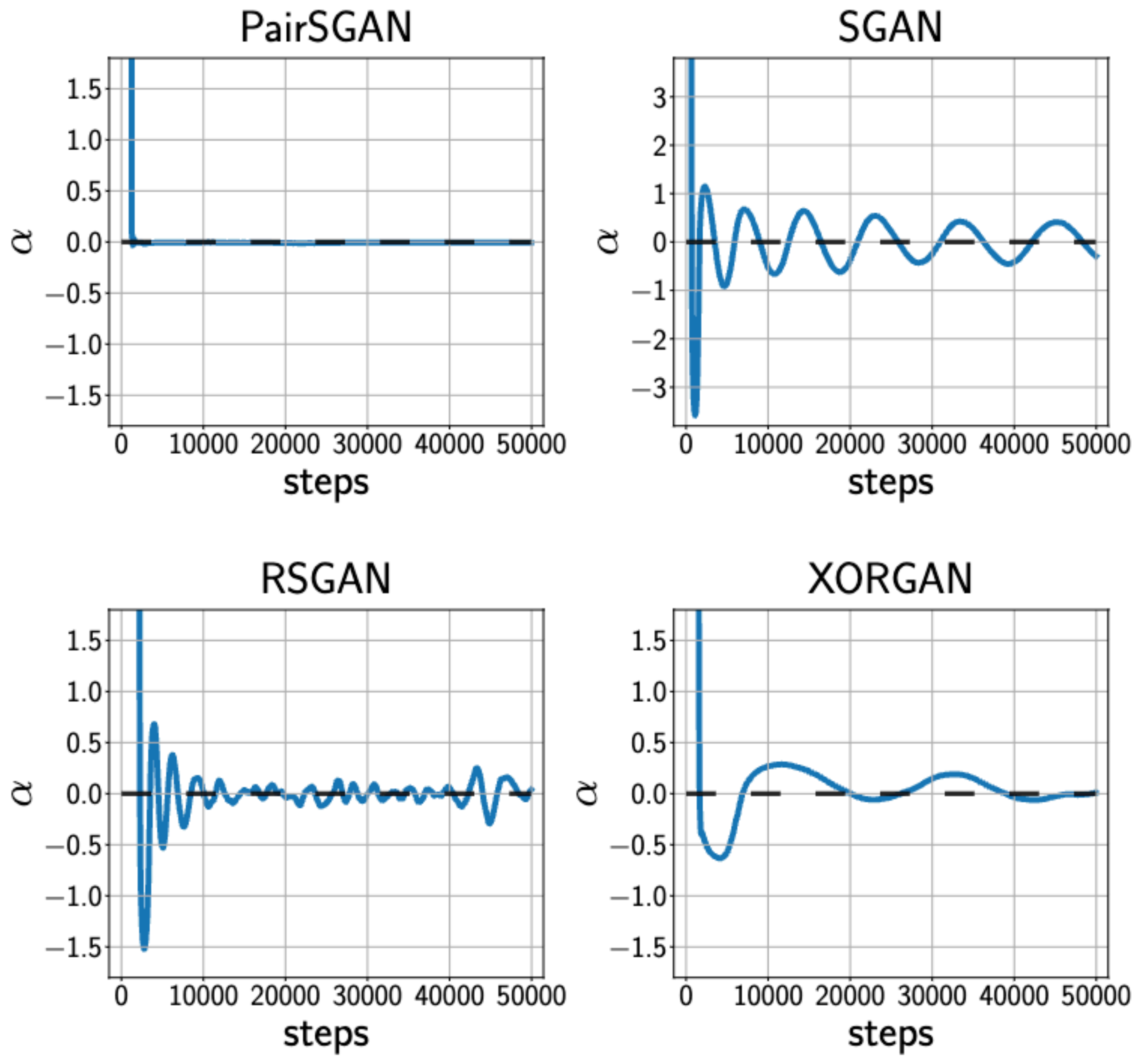
Alignment stability in
restricted DCGAN generator experiment



Experimental validation

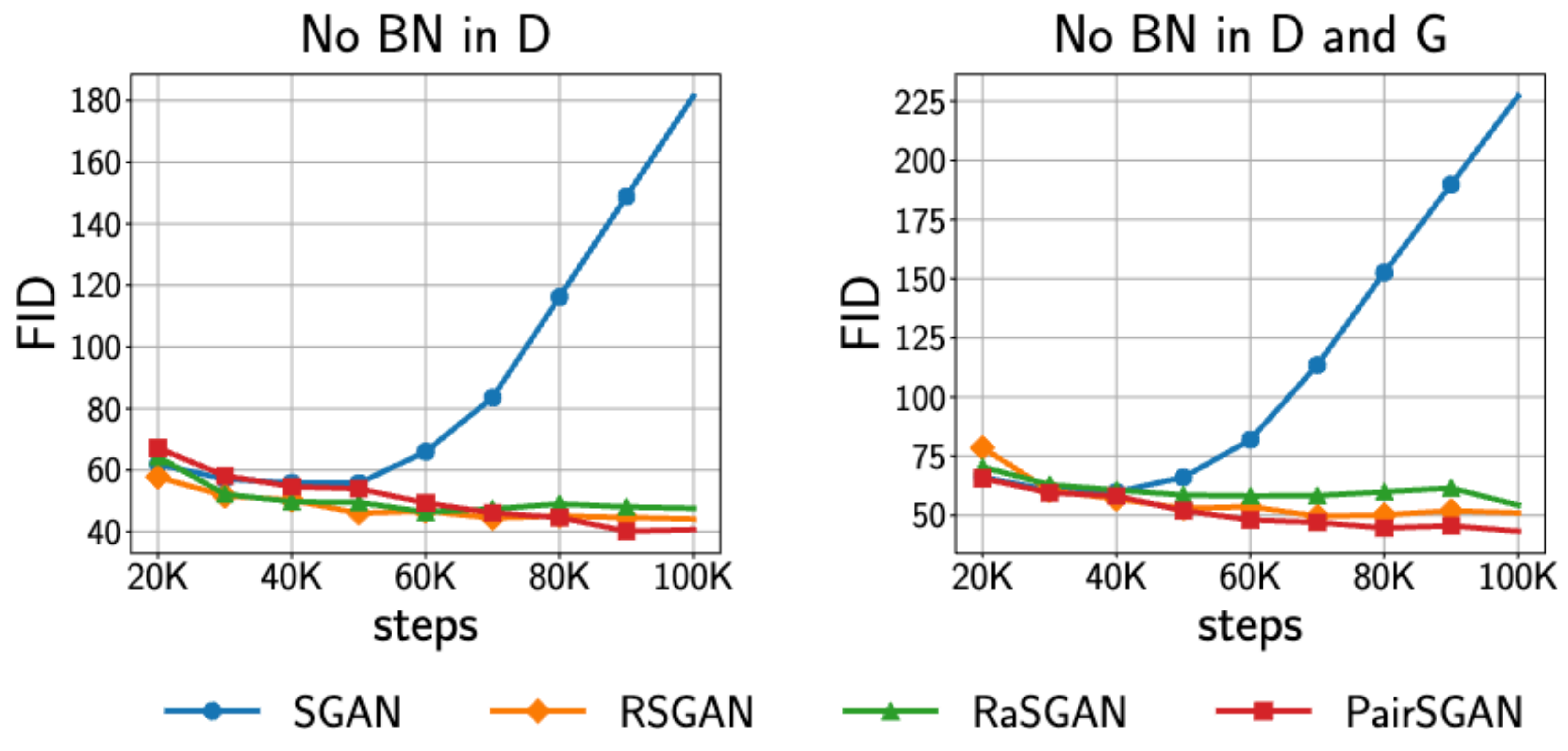
- Verified alignment preservation in DCGAN with restricted generator
- Improved FID curve stability on CIFAR-10 and CAT benchmarks

Alignment stability in restricted DCGAN generator experiment



Improved FID on CIFAR 10 without BN in D / G&D

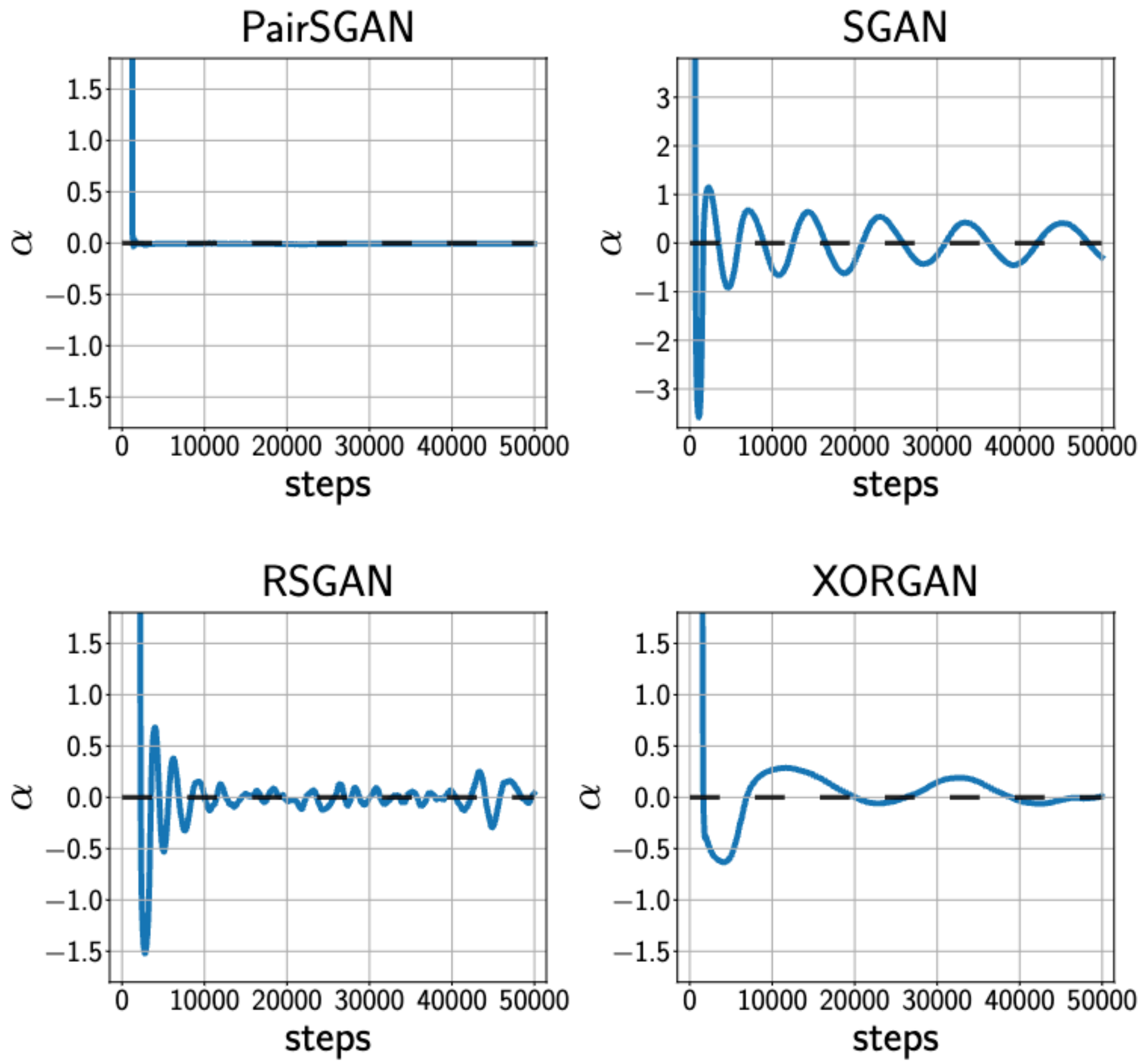
Loss	D	G & D
SGAN	181.47	227.17
RSGAN	44.14	51.00
RaSGAN	47.63	54.28
PairSGAN	**40.13**	**43.24**



Experimental validation

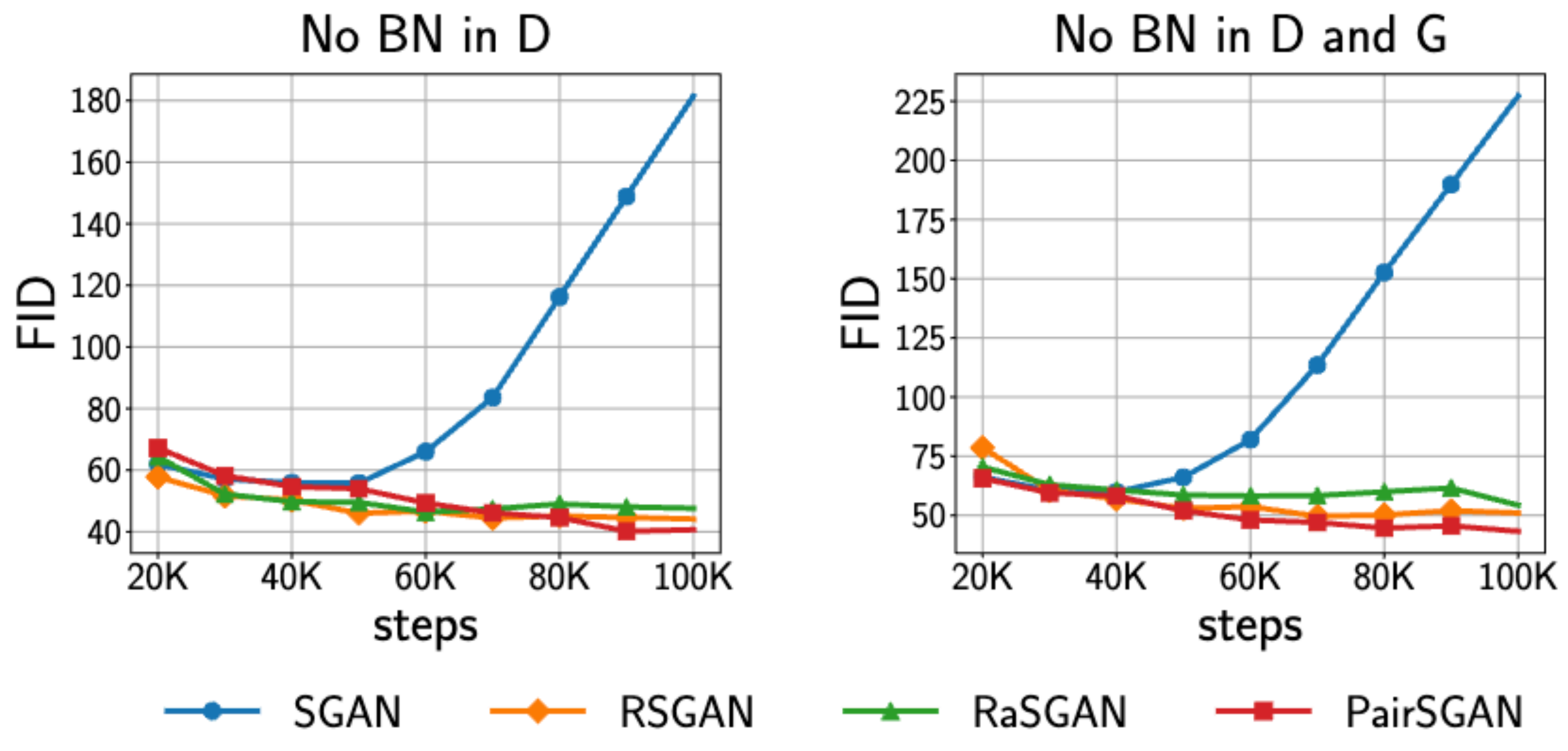
- Verified alignment preservation in DCGAN with restricted generator
- Improved FID curve stability on CIFAR-10 and CAT benchmarks

Alignment stability in restricted DCGAN generator experiment



Improved FID on CIFAR 10 without BN in D / G&D

Loss	D	G & D
SGAN	181.47	227.17
RSGAN	44.14	51.00
RaSGAN	47.63	54.28
PairSGAN	**40.13**	**43.24**



Stable FID curves on CAT data

Loss	Min	Max	Mean	SD
64 × 64 images				
SGAN	12.27	24.62	16.99	4.07
RSGAN*	19.03	42.05	32.16	7.01
RaSGAN*	15.38	33.11	20.53	5.68
LSGAN*	20.27	224.97	73.62	61.02
RaLSGAN*	11.97	19.29	15.61	2.55
HingeGAN*	17.60	50.94	32.23	14.44
RaHingeGAN*	14.62	27.31	20.29	3.96
RSGAN-GP*	16.41	22.34	18.20	1.82
RaSGAN-GP*	17.32	22	19.58	**1.81**
PairSGAN	**10.28**	**18.21**	**13.55**	2.24
128 × 128 images				
SGAN	19.88	38.68	28.91	6.73
RaSGAN*	21.05	39.65	28.53	6.52
LSGAN*	19.03	51.36	30.28	10.16
RaLSGAN*	**15.85**	40.26	22.36	7.53
PairSGAN	16.72	**25.66**	**21.43**	**2.94**
256 × 256 images				
SGAN	43.30	324.38	171.42	108.47
RaSGAN*	**32.11**	102.76	56.64	21.03
SpectralSGAN*	54.08	90.43	64.92	12.00
LSGAN*	—	—	—	—
RaLSGAN*	35.21	299.52	70.44	86.01
WGAN-GP*	155.46	437.48	341.91	101.11
PairSGAN	33.94	**50.52**	**41.70**	**5.23**

Pairwise-Discriminator Objectives for GANs: Summary

Training objectives for GANs which **preserve distribution alignment**

- Novel zero-sum & non-zero-sum game formulations with pairwise discriminators
  - New distribution divergences derived from PairGAN game
- Equilibria stability analysis for parametric generators
  - Introduced the notion of sufficient discriminators for given parametric generator family
  - Constructed examples of minimally sufficient discriminators for arbitrary generator families
  - Established connections to non-parametric MMD objectives and MMD-GAN
- Extension to multiple distribution alignment scenario

Experimental validation

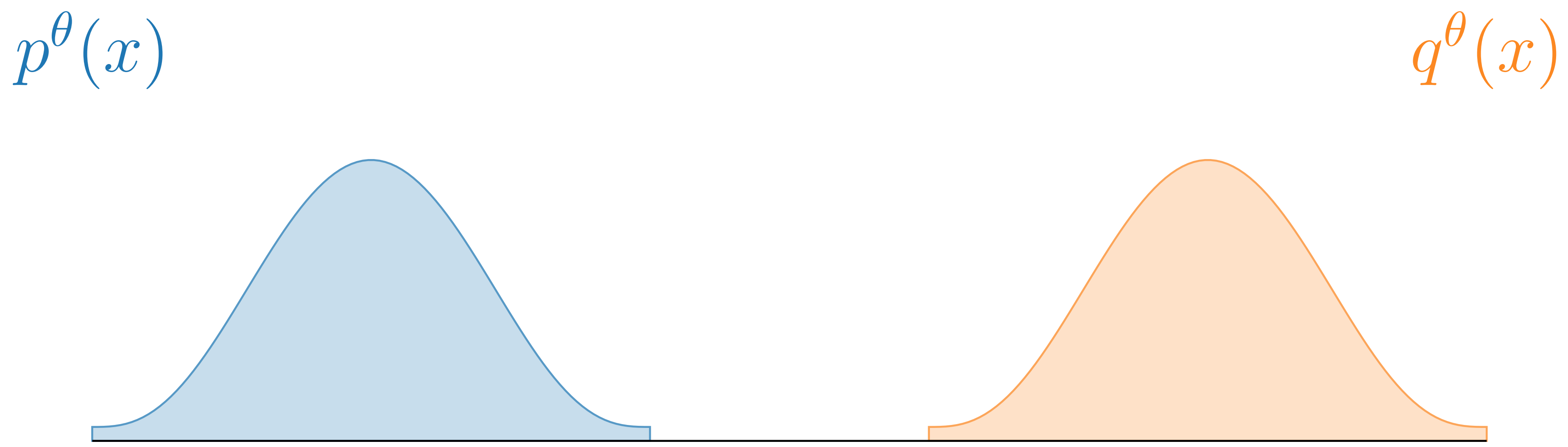
- Demonstrated alignment preserving property in DCGAN under restricted generator parameterization
- Improved FID curve stability on CIFAR-10 and CAT benchmarks

Chapter III
Adversarial Support Alignment

Distribution Alignment

Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\}$ $\mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

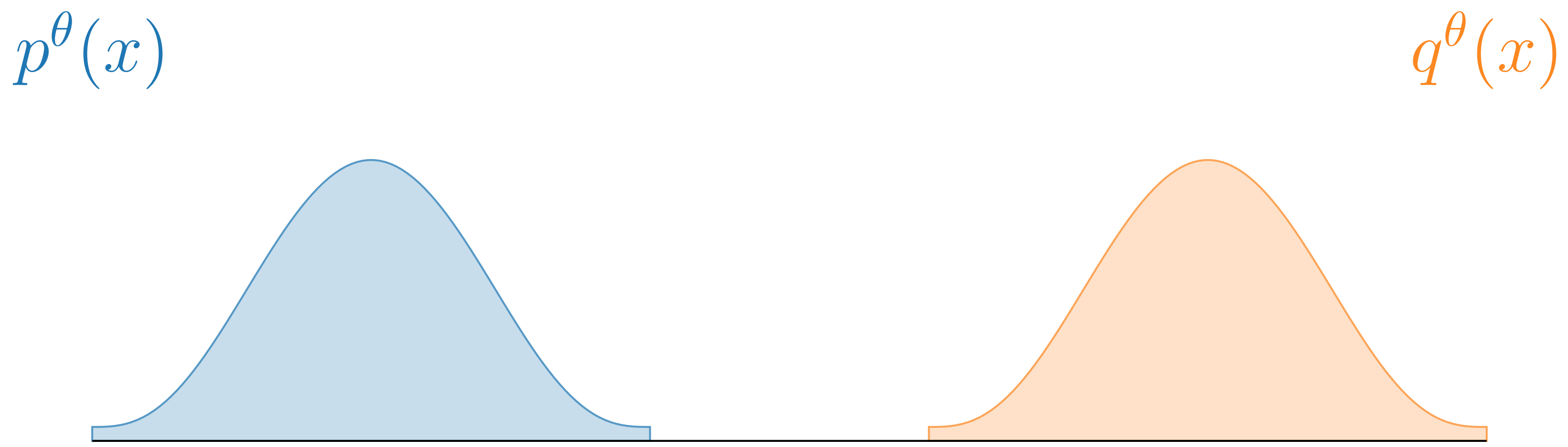
Find $\theta^* : \; p^{\theta^*} = q^{\theta^*}$

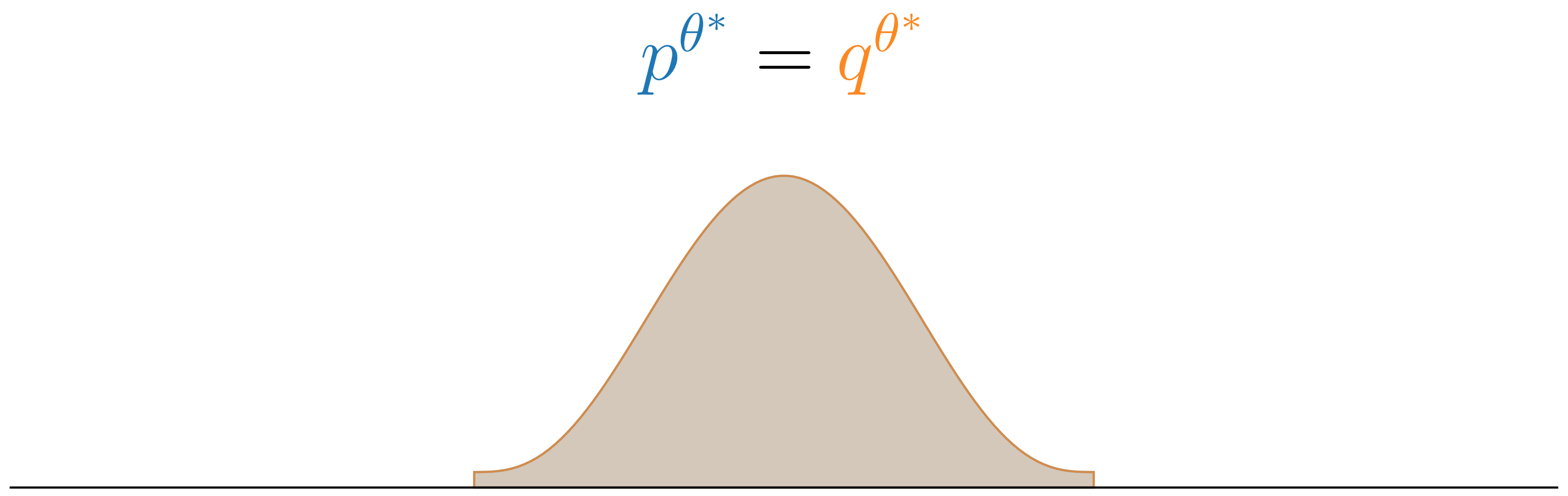


Distribution Alignment

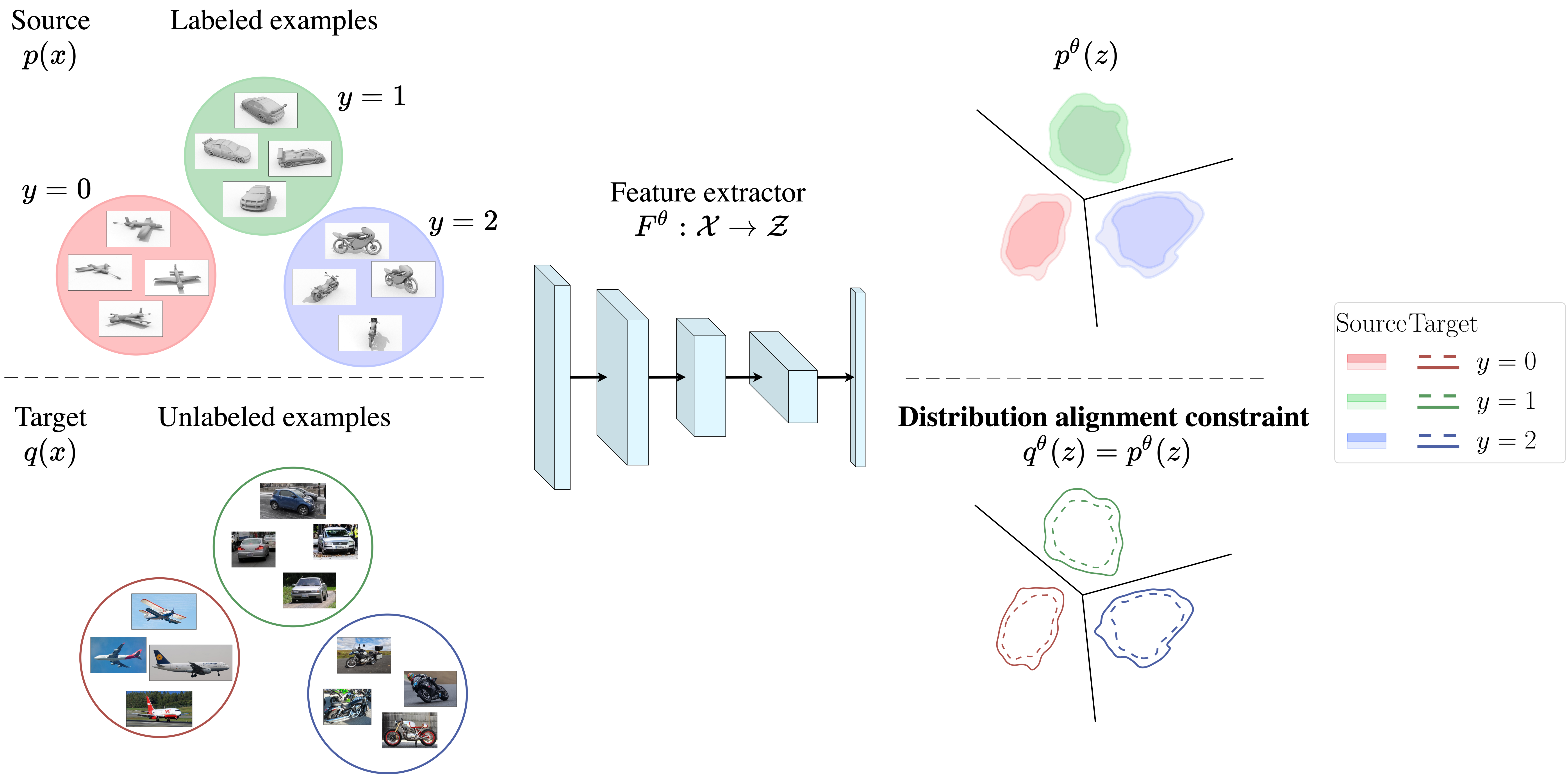
Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\}$ $\quad$ $\mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

Find $\theta^* : \; p^{\theta^*} = q^{\theta^*}$





Distribution Alignment for Domain Adaptation

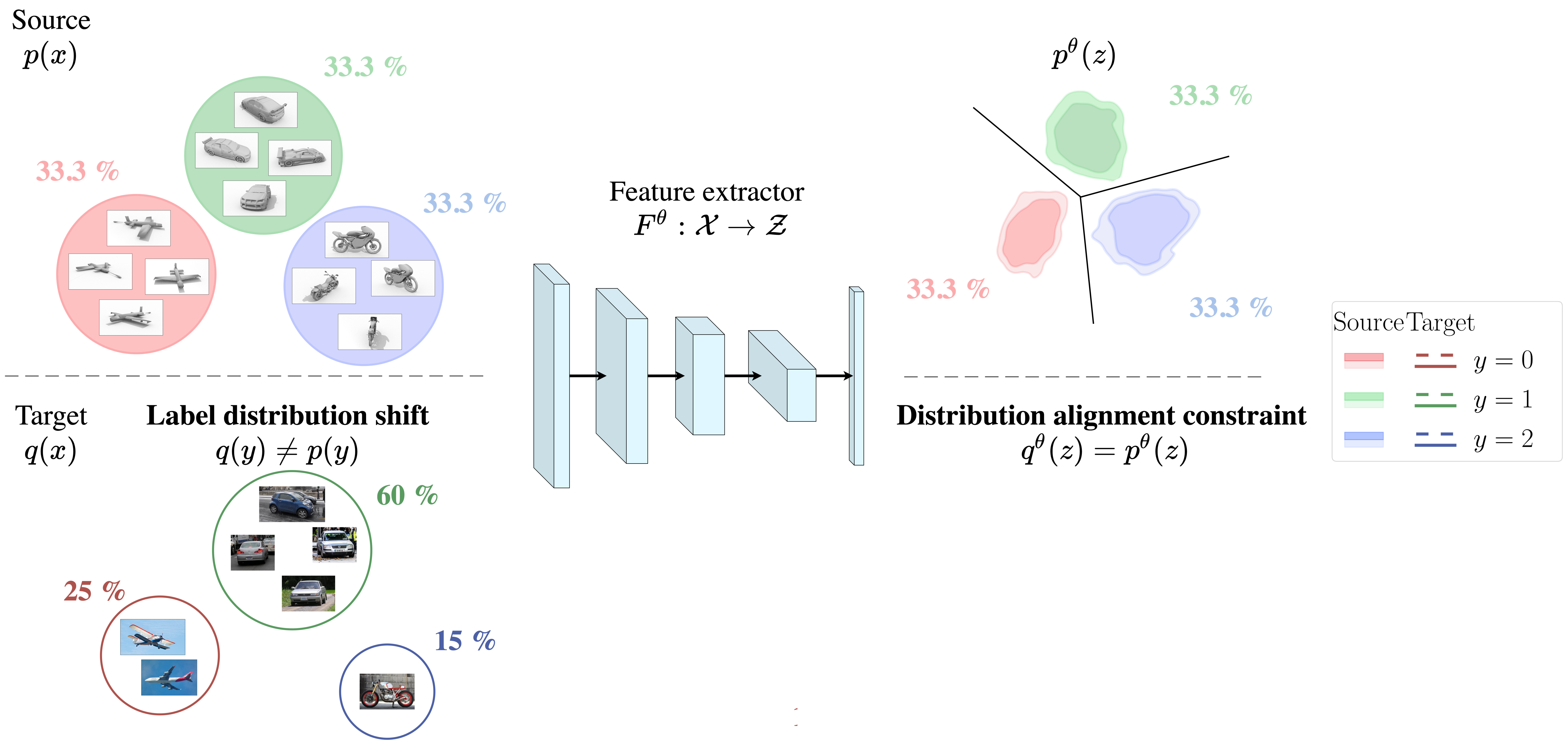


Two training objectives:

- Source domain classification loss
- Source vs target discrimination loss (GAN-like game with a discriminator)

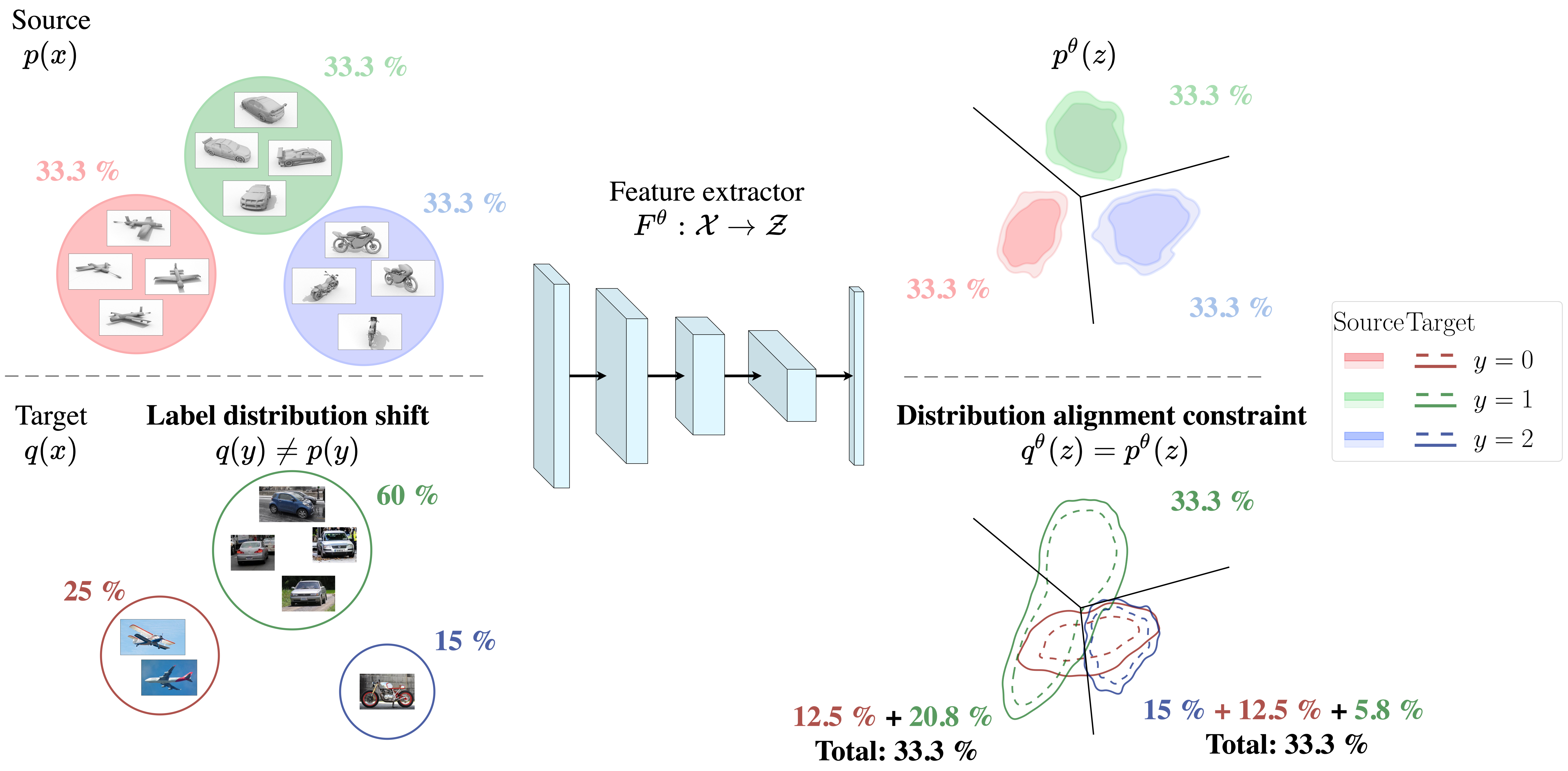
Domain Adversarial Neural Networks [Ganin et al., 2016]

Issue: Label Distribution Shift



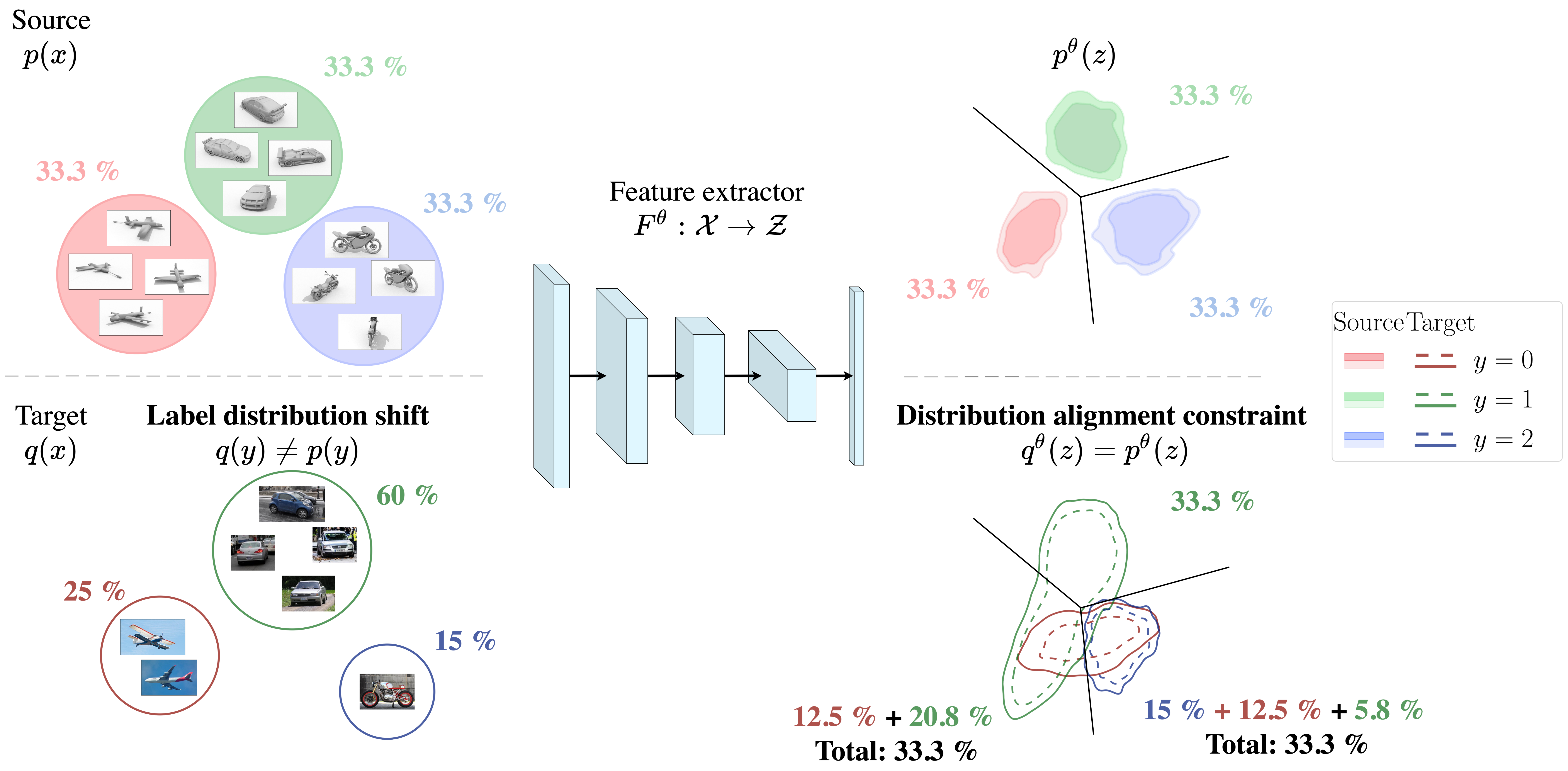
[Zhao et al., 2019; Wu et al., 2019; Tachet des Combes et al, 2020, …]

Issue: Label Distribution Shift



[Zhao et al., 2019; Wu et al., 2019; Tachet des Combes et al, 2020, …]

Issue: Label Distribution Shift



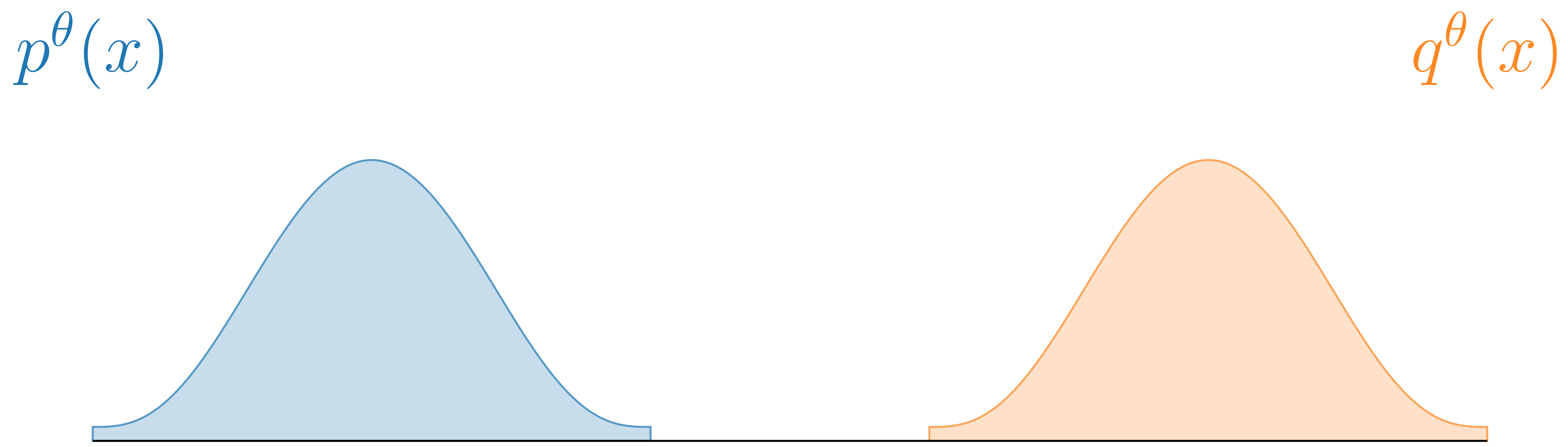
strict distribution alignment → source-target class mismatch → degraded accuracy

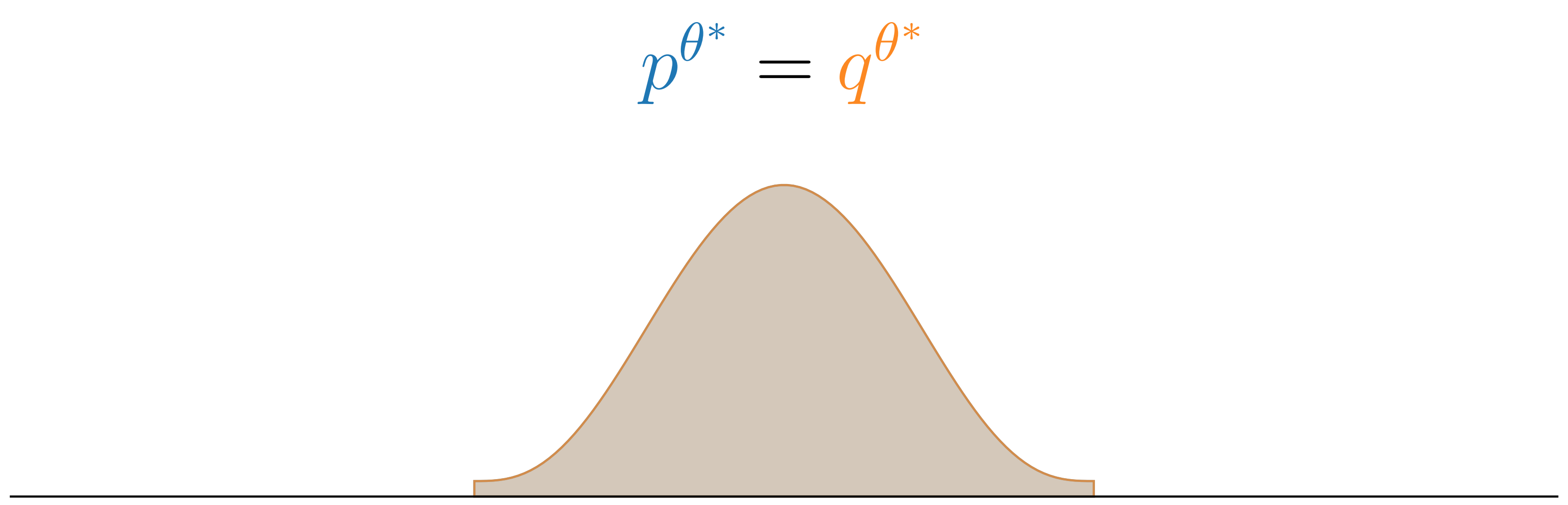
[Zhao et al., 2019; Wu et al., 2019; Tachet des Combes et al, 2020, …]

Distribution Alignment

Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\}$ $\quad \mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

Find $\theta^* : \; p^{\theta^*} = q^{\theta^*}$



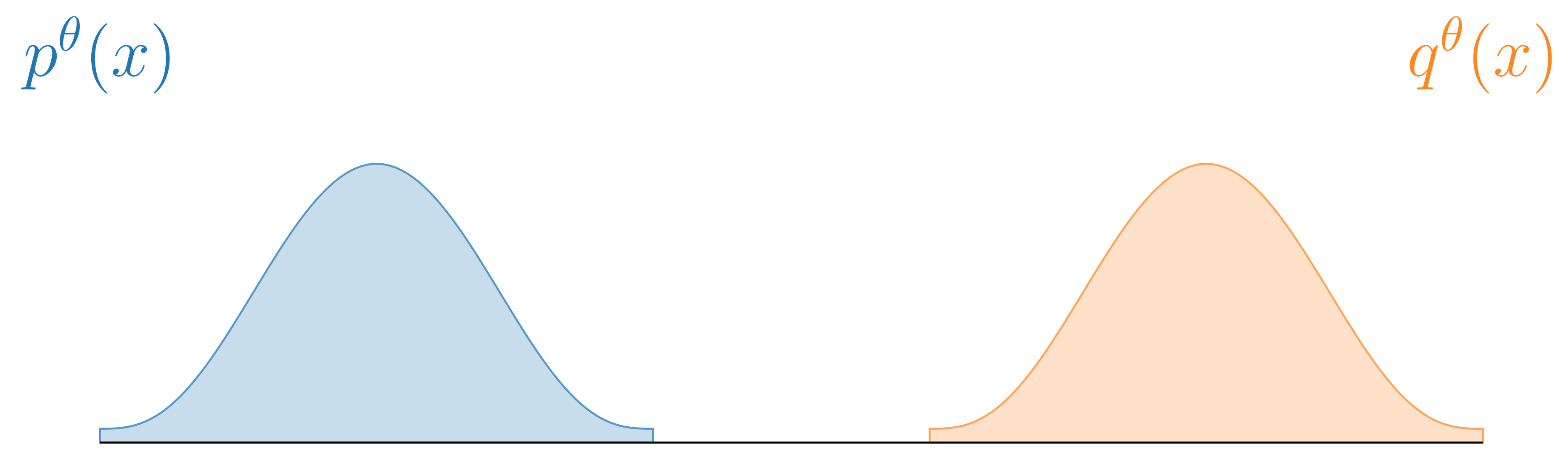


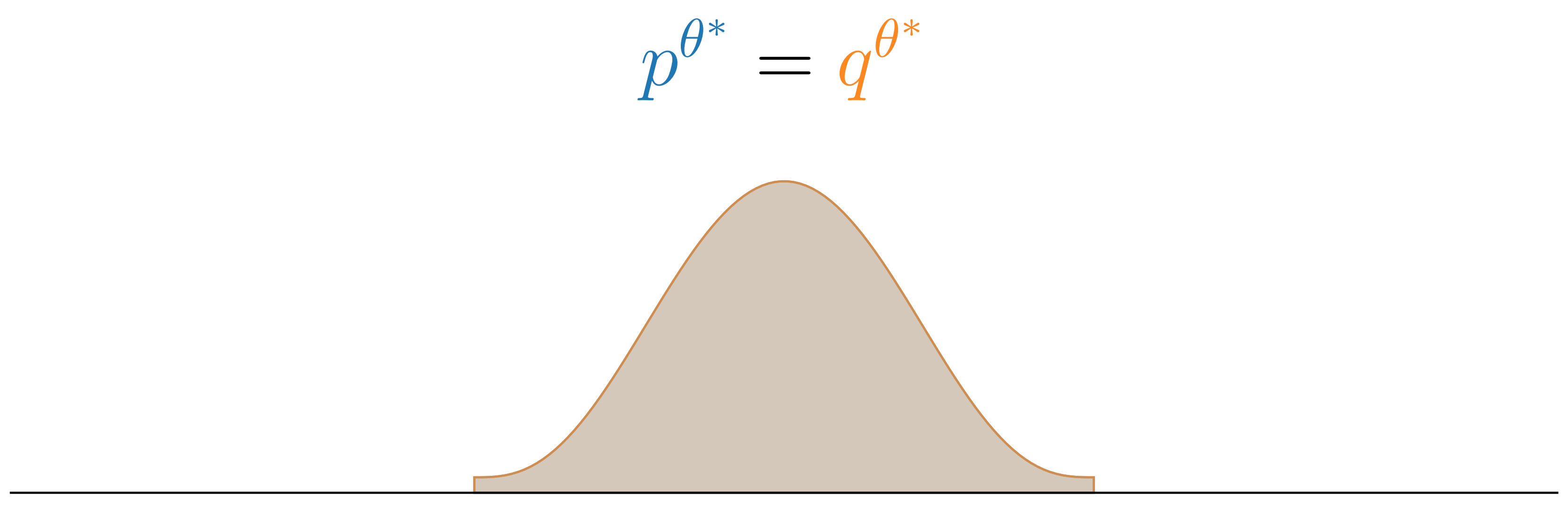
How can we lift constraints of strict distribution alignment?

Distribution Alignment

Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\} \quad \mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

Find $\theta^* : \; p^{\theta^*} = q^{\theta^*}$



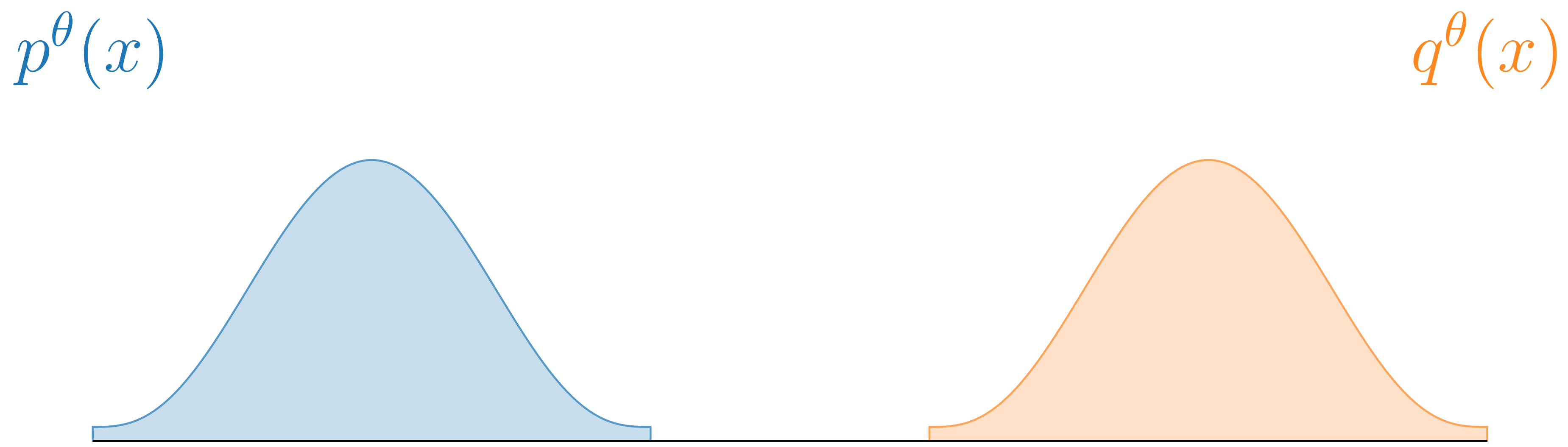


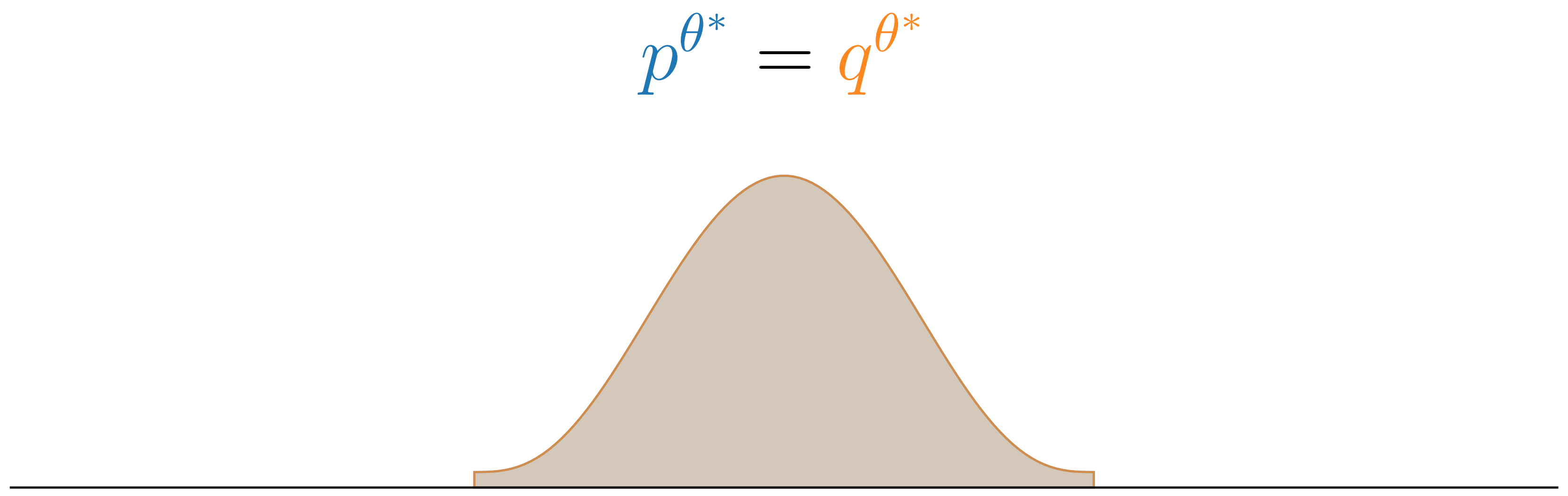
How can we lift constraints of strict distribution alignment?

Distribution Alignment

Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\}$ $\mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

Find $\theta^* : \ p^{\theta^*} = q^{\theta^*}$





Support Alignment

Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\}$ $\mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

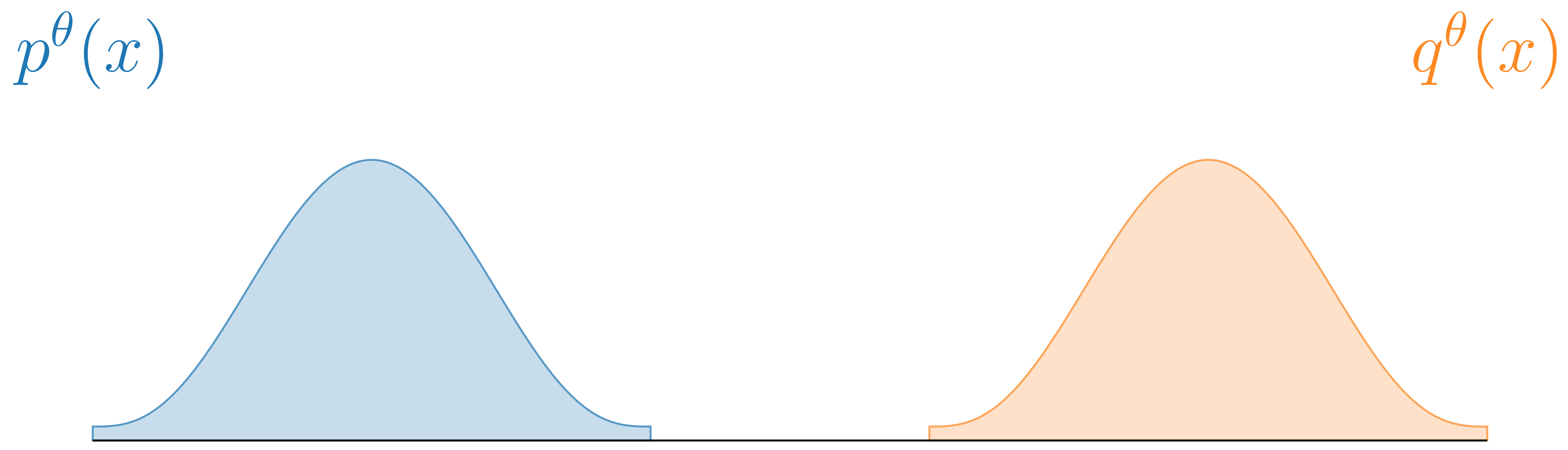
Find $\theta^* : \ \mathrm{supp}(p^{\theta^*}) = \mathrm{supp}(q^{\theta^*})$

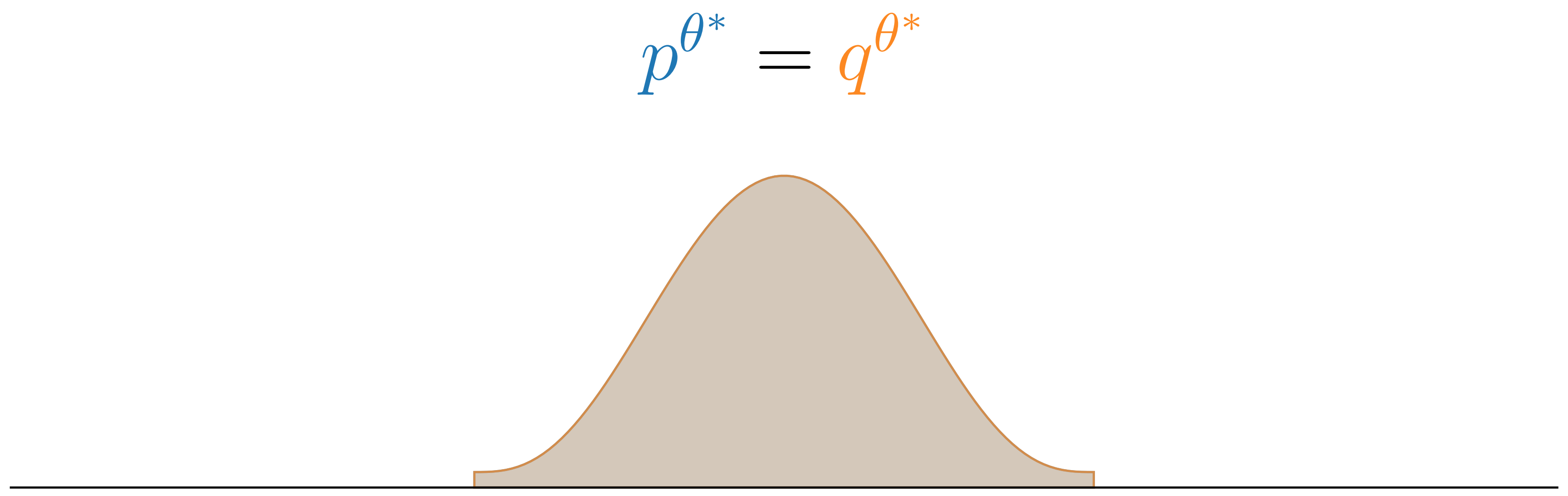
How can we lift constraints of strict distribution alignment?

Distribution Alignment

Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\}$ $\quad \mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

Find $\theta^* : \; p^{\theta^*} = q^{\theta^*}$

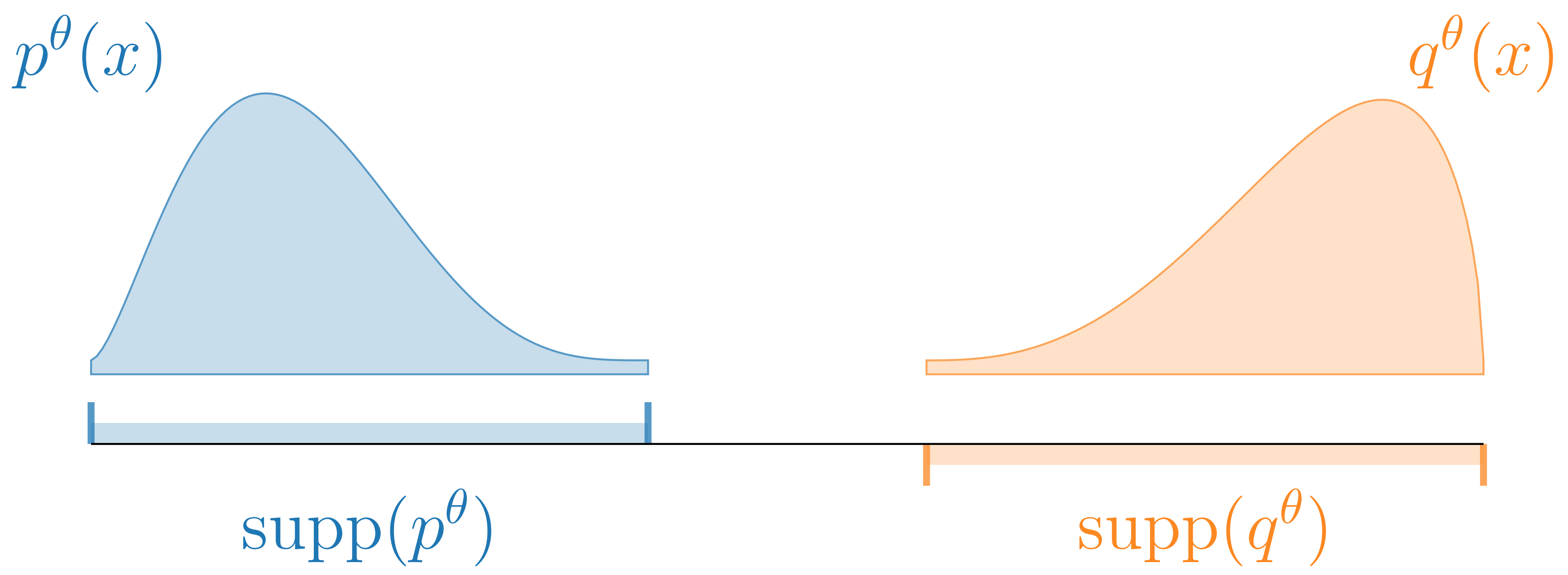




Support Alignment

Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\}$ $\quad \mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

Find $\theta^* : \; \mathrm{supp}(p^{\theta^*}) = \mathrm{supp}(q^{\theta^*})$

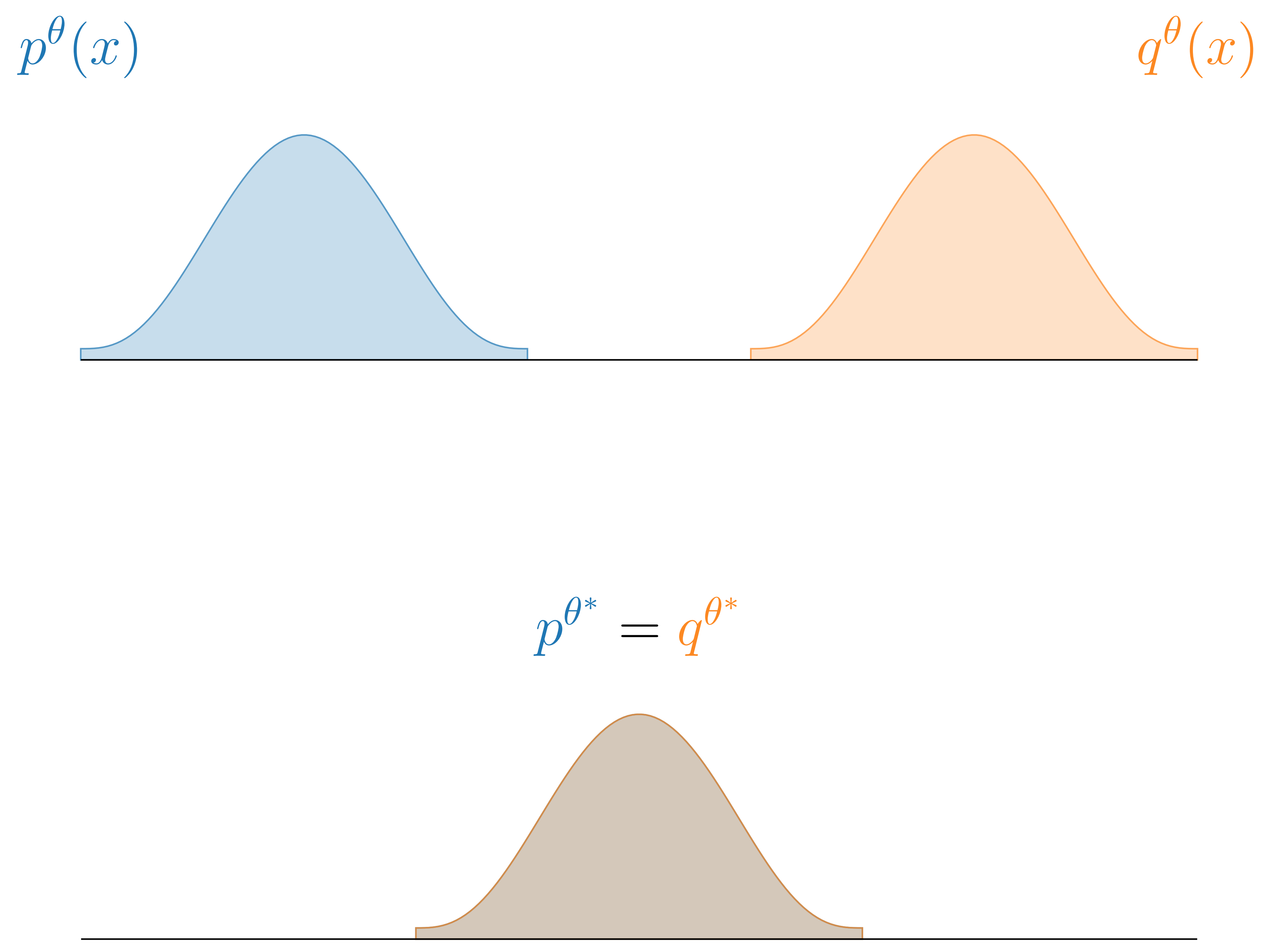


How can we lift constraints of strict distribution alignment?

Distribution Alignment

Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\}$ $\quad \mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

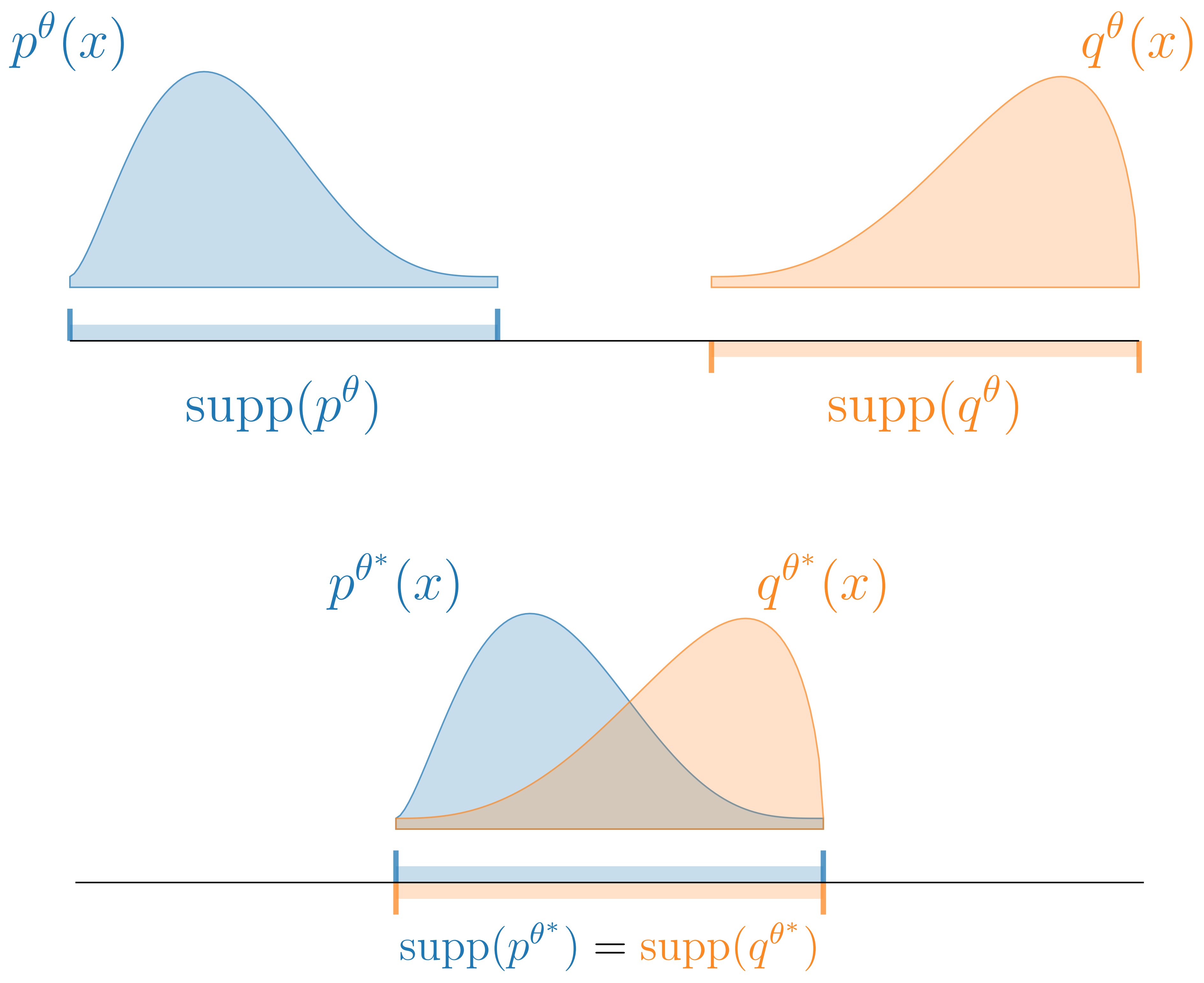
Find $\theta^* : \; p^{\theta^*} = q^{\theta^*}$



Support Alignment

Given $\mathcal{P} = \{p^\theta \mid \theta \in \Theta\}$ $\quad \mathcal{Q} = \{q^\theta \mid \theta \in \Theta\}$

Find $\theta^* : \; \mathrm{supp}(p^{\theta^*}) = \mathrm{supp}(q^{\theta^*})$



Method: Support Alignment via Log-Loss Discriminator

$$\sup_{f:\mathcal{X}\to[0,1]} \mathop{\mathbb{E}}_{x\sim p}[\log f(x)] + \mathop{\mathbb{E}}_{y\sim q}[\log(1-f(y))]$$

Method: Support Alignment via Log-Loss Discriminator

$$\sup_{f:\mathcal{X}\to[0,1]} \mathop{\mathbb{E}}_{x\sim p}[\log f(x)] + \mathop{\mathbb{E}}_{y\sim q}[\log(1-f(y))]$$

$$f^*(x) = \frac{p(x)}{p(x)+q(x)}$$

Method: Support Alignment via Log-Loss Discriminator

$$\sup_{f:\mathcal{X}\to[0,1]} \mathop{\mathbb{E}}_{x\sim p}[\log f(x)] + \mathop{\mathbb{E}}_{y\sim q}[\log(1-f(y))]$$

$$f^*(x) = \frac{p(x)}{p(x)+q(x)}$$

Theorem

The mapping $f^* : \mathcal{X} \to [0,1]$
realized by the optimal discriminator
preserves support discrepancy

$$\mathrm{supp}(p) = \mathrm{supp}(q) \Leftrightarrow \mathrm{supp}(f^*{}_\sharp p) = \mathrm{supp}(f^*{}_\sharp q)$$

$f^*{}_\sharp p,\ f^*{}_\sharp q$ — pushforward distributions

Method: Support Alignment via Log-Loss Discriminator

$$\sup_{f:\mathcal{X}\to[0,1]} \mathop{\mathbb{E}}_{x\sim p}[\log f(x)] + \mathop{\mathbb{E}}_{y\sim q}[\log(1-f(y))]$$

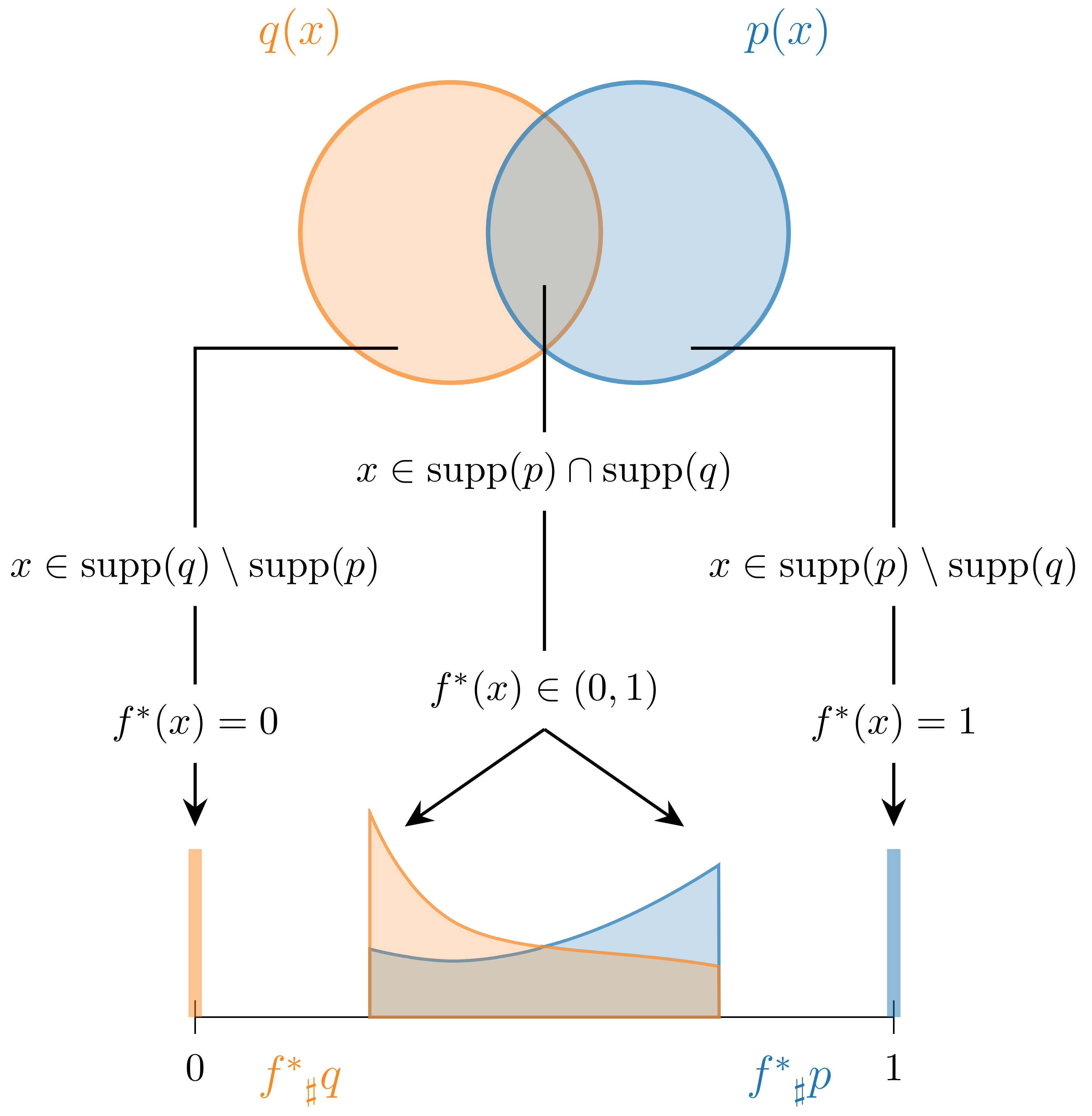
$$f^*(x) = \frac{p(x)}{p(x)+q(x)}$$

Theorem

The mapping $f^* : \mathcal{X} \to [0,1]$
realized by the optimal discriminator
preserves support discrepancy

$\mathrm{supp}(p) = \mathrm{supp}(q) \Leftrightarrow \mathrm{supp}(f^*{}_\sharp p) = \mathrm{supp}(f^*{}_\sharp q)$

$f^*{}_\sharp p,\ f^*{}_\sharp q$ — pushforward distributions



Method: Support Alignment via Log-Loss Discriminator

$$\sup_{f:\mathcal{X}\to[0,1]} \mathop{\mathbb{E}}_{x\sim p}[\log f(x)] + \mathop{\mathbb{E}}_{y\sim q}[\log(1-f(y))]$$

$$f^*(x) = \frac{p(x)}{p(x)+q(x)}$$

Theorem

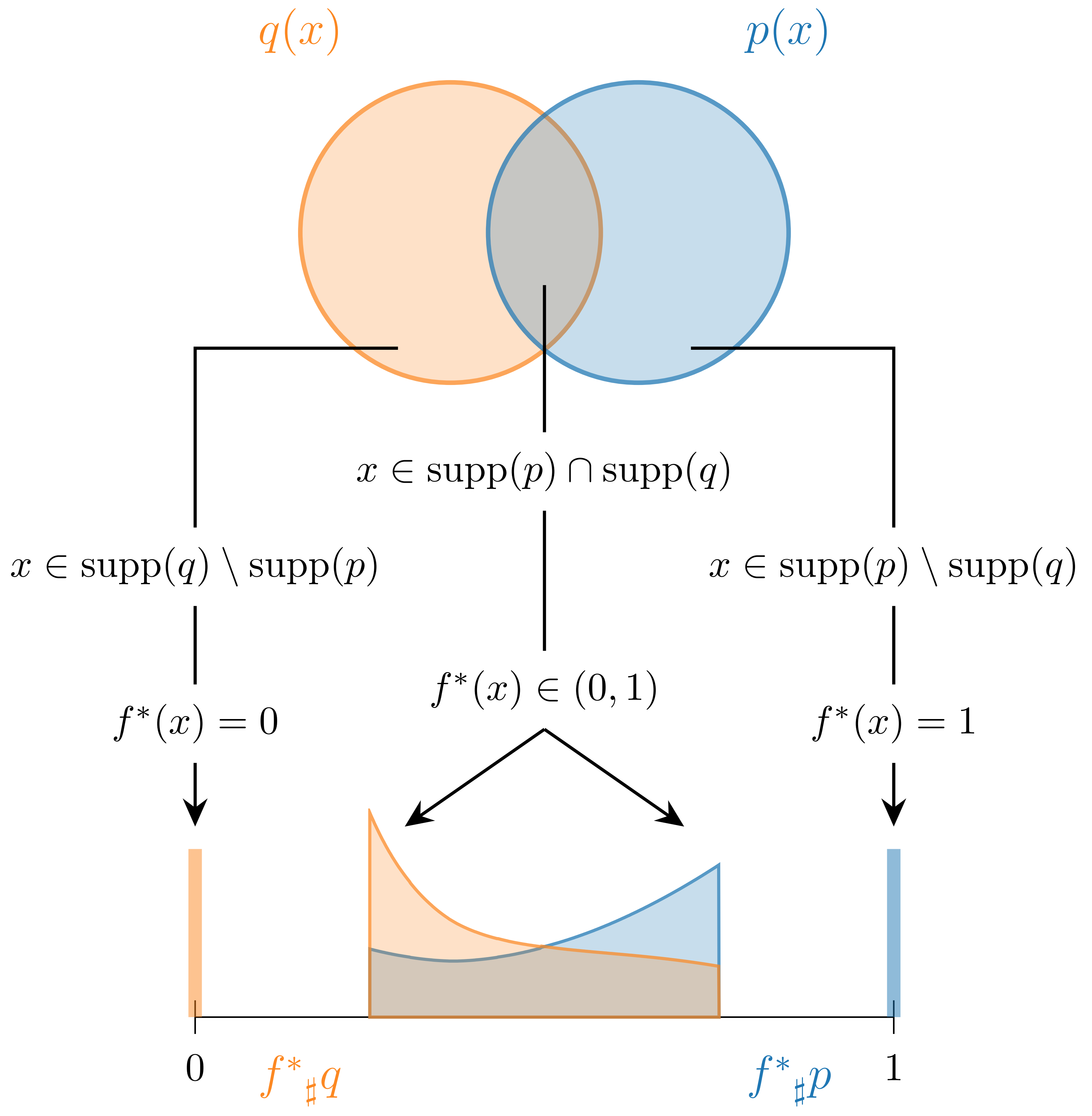
The mapping $f^* : \mathcal{X} \to [0,1]$
realized by the optimal discriminator
preserves support discrepancy

$$\mathrm{supp}(p) = \mathrm{supp}(q) \Leftrightarrow \mathrm{supp}(f^*{}_\sharp p) = \mathrm{supp}(f^*{}_\sharp q)$$

${f^*}_\sharp p,\ {f^*}_\sharp q$ — pushforward distributions

Remark: the result also holds for $g : \mathcal{X} \to \mathbb{R}$

$g(x) : f(x) = \mathrm{sigmoid}(g(x))$



Method: Support Alignment via Log-Loss Discriminator

$$\sup_{f:\mathcal{X}\to[0,1]} \mathop{\mathbb{E}}_{x\sim p}[\log f(x)] + \mathop{\mathbb{E}}_{y\sim q}[\log(1-f(y))]$$

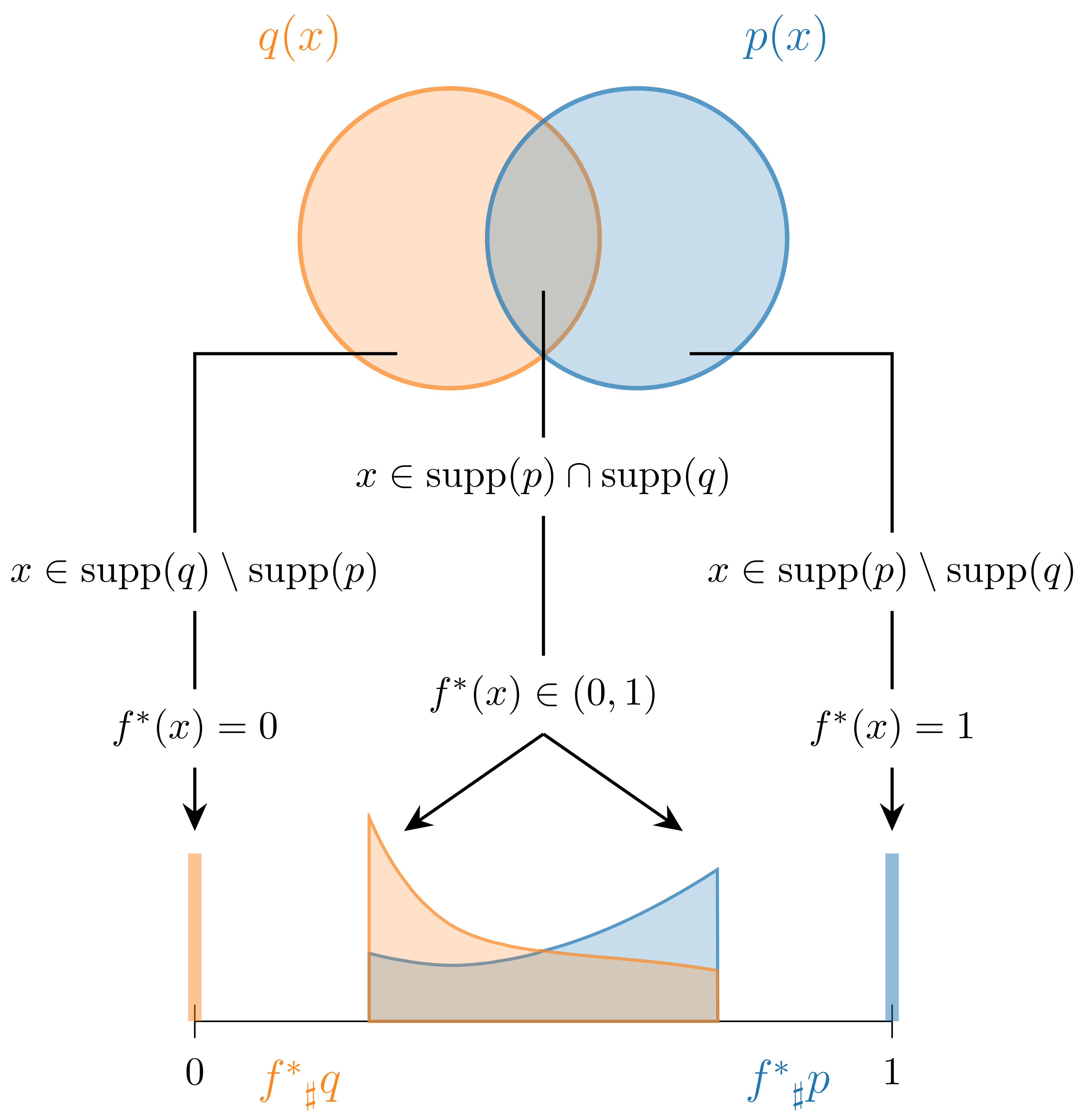
$$f^*(x) = \frac{p(x)}{p(x)+q(x)}$$

Theorem

The mapping $f^* : \mathcal{X} \to [0,1]$
realized by the optimal discriminator
preserves support discrepancy

$$\text{supp}(p) = \text{supp}(q) \Leftrightarrow \text{supp}(f^*{}_\sharp p) = \text{supp}(f^*{}_\sharp q)$$

$f^*{}_\sharp p,\ f^*{}_\sharp q$ — pushforward distributions



Remark: the result also holds for $g : \mathcal{X} \to \mathbb{R}$

$g(x) : f(x) = \text{sigmoid}(g(x))$

Not all discriminators classes have this property (details in the paper)

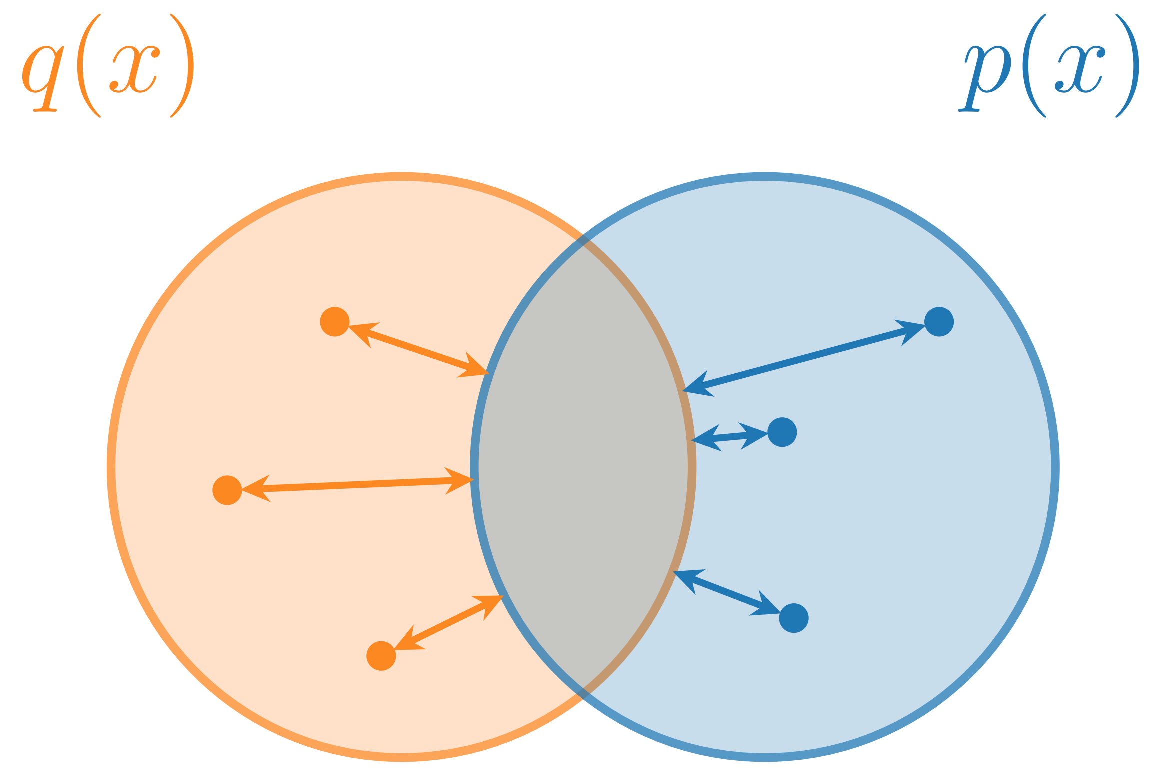
Support Difference

Symmetric Support Difference* / Chamfer Distance

$$\mathcal{D}_{\triangle}(p,q) = \mathbb{E}_{x^q \sim q}\Big[d(x^q, \mathrm{supp}(p))\Big] + \mathbb{E}_{x^p \sim p}\Big[d(x^p, \mathrm{supp}(q))\Big]$$

$$d(x^q, \mathrm{supp}(p)) = \inf_{x^p \in \mathrm{supp}(p)} d(x^q, x^p)$$

$$d(x^p, \mathrm{supp}(q)) = \inf_{x^q \in \mathrm{supp}(q)} d(x^p, x^q)$$



$$1)\ \ \mathcal{D}_{\triangle}(p,q) \geq 0 \ \ \forall\, p, q; \qquad 2)\ \ \mathcal{D}_{\triangle}(p,q) = 0 \iff \mathrm{supp}(p) = \mathrm{supp}(q)$$

*generalizes Chamfer distance to continuous distributions

Spectrum of Alignment Criteria

Wasserstein distance

$$\mathcal{D}_W(p, q) = \inf_{\gamma} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} \left[d(x, y) \right]$$

$$\text{s.t} \int \gamma(x, y)\, dy = p(x)$$

$$\int \gamma(x, y)\, dx = q(y)$$

$$\mathcal{D}_W(p, q) = 0 \iff p = q$$

Spectrum of Alignment Criteria

Wasserstein distance

$$\mathcal{D}_W(p,q) = \inf_{\gamma} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} \left[d(x,y)\right]$$

$$\text{s.t} \int \gamma(x,y)\,dy = p(x)$$

$$\int \gamma(x,y)\,dx = q(y)$$

$$\mathcal{D}_W(p,q) = 0 \iff p = q$$

β-admissible Wasserstein distance $(\beta > 0)$ [Wu et al., 2019]

$$\mathcal{D}_W^{\beta}(p,q) = \inf_{\gamma} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} \left[d(x,y)\right]$$

$$\text{s.t.} \int \gamma(x,y)\,dy = p(x)$$

$$\int \gamma(x,y)\,dx \leq (1+\beta)q(y)$$

$$\mathcal{D}_W^{\beta}(p,q) = 0 \iff p(x) \leq (1+\beta)q(x)$$

Spectrum of Alignment Criteria

Wasserstein distance

$$\mathcal{D}_W(p, q) = \inf_{\gamma} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} \big[d(x, y)\big]$$
$$\text{s.t} \int \gamma(x, y)\, dy = p(x)$$
$$\int \gamma(x, y)\, dx = q(y)$$

$$\mathcal{D}_W(p, q) = 0 \iff p = q$$

β-admissible Wasserstein distance $(\beta > 0)$ [Wu et al., 2019]

$$\mathcal{D}_W^{\beta}(p, q) = \inf_{\gamma} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} \big[d(x, y)\big]$$
$$\text{s.t.} \int \gamma(x, y)\, dy = p(x)$$
$$\int \gamma(x, y)\, dx \le (1+\beta) q(y)$$

$$\mathcal{D}_W^{\beta}(p, q) = 0 \iff p(x) \le (1+\beta) q(x)$$

Support Difference (SD)

$$\mathcal{D}_W^{\infty}(p, q) = \inf_{\gamma} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} \big[d(x, y)\big] \qquad = \mathop{\mathbb{E}}_{x\sim p(x)} \big[\inf_{y\in \mathrm{supp}(q)} d(x, y)\big]$$
$$\text{s.t.} \int \gamma(x, y)\, dy = p(x)$$
$$q(y) = 0 \implies \int \gamma(x, y)\, dx = 0$$

$$\mathcal{D}_W^{\infty}(p, q) = 0 \iff \mathrm{supp}(p) \subset \mathrm{supp}(q)$$

Spectrum of Alignment Criteria

Wasserstein distance

$$\mathcal{D}_W(p,q) = \inf_{\gamma} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} \left[d(x,y)\right]$$
$$\text{s.t} \int \gamma(x,y)\,dy = p(x)$$
$$\int \gamma(x,y)\,dx = q(y)$$

$$\mathcal{D}_W(p,q) = 0 \iff p = q$$

β-admissible Wasserstein distance $(\beta > 0)$ [Wu et al., 2019]

$$\mathcal{D}_W^{\beta}(p,q) = \inf_{\gamma} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} \left[d(x,y)\right]$$
$$\text{s.t.} \int \gamma(x,y)\,dy = p(x)$$
$$\int \gamma(x,y)\,dx \le (1+\beta)q(y)$$

$$\mathcal{D}_W^{\beta}(p,q) = 0 \iff p(x) \le (1+\beta)q(x)$$

Support Difference (SD)

$$\mathcal{D}_W^{\infty}(p,q) = \inf_{\gamma} \mathop{\mathbb{E}}_{(x,y)\sim\gamma} \left[d(x,y)\right] \qquad = \mathop{\mathbb{E}}_{x\sim p(x)} \left[\inf_{y\in \mathrm{supp}(q)} d(x,y)\right]$$
$$\text{s.t.} \int \gamma(x,y)\,dy = p(x)$$
$$q(y) = 0 \implies \int \gamma(x,y)\,dx = 0$$

$$\mathcal{D}_W^{\infty}(p,q) = 0 \iff \mathrm{supp}(p) \subset \mathrm{supp}(q)$$

$$\mathcal{D}_{\triangle}(p,q) = \mathcal{D}_W^{\infty}(p,q) + \mathcal{D}_W^{\infty}(q,p) = \lim_{\beta\to\infty} \mathcal{D}_W^{\beta}(p,q) + \mathcal{D}_W^{\beta}(q,p)$$

$$\mathcal{D}_{\triangle}(p,q) = 0 \iff \mathrm{supp}(p) = \mathrm{supp}(q)$$

Spectrum of Alignment Criteria

Proposition (Alignment Conditions are Preserved by Log-loss Discriminator)**.**
Let f^* *be the optimal discriminator* $f^*(x) = \frac{p(x)}{p(x)+q(x)}$ *for distributions* p*,* q*.*
Let $\mathcal{D}_W^{\beta_1,\beta_2}(p,q) = \mathcal{D}_W^{\beta_1}(p,q) + \mathcal{D}_W^{\beta_2}(q,p)$*. Then,*

1. $\mathcal{D}_W(p,q) = 0$ *if and only if* $\mathcal{D}_W({f^*}_\sharp p, {f^*}_\sharp q) = 0$*; (distribution alignment)*
2. $\mathcal{D}_W^{\beta_1,\beta_2}(p,q) = 0$ *if and only if* $\mathcal{D}_W^{\beta_1,\beta_2}({f^*}_\sharp p, {f^*}_\sharp q) = 0$*; (relaxed distribution alignment)*
3. $\mathcal{D}_\triangle(p,q) = 0$ *if and only if* $\mathcal{D}_\triangle({f^*}_\sharp p, {f^*}_\sharp q) = 0$*. (support alignment)*

Adversarial Support Alignment

$$\mathcal{L}_D(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{g(x)}\right)\right]$$

Adversarial Support Alignment

$$\mathcal{L}_D(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{g(x)}\right)\right]$$

Alignment objective: $\min_{\theta} \; \mathcal{L}_A(\theta, g)$

Adversarial Support Alignment

$$\mathcal{L}_D(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{g(x)}\right)\right]$$

Alignment objective: $\min_{\theta} \; \mathcal{L}_A(\theta, g)$

Adversarial Distribution Alignment (GAN/DANN)

$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right]$$

Adversarial Support Alignment

$$\mathcal{L}_D(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{g(x)}\right)\right]$$

Alignment objective: $\min_\theta \ \mathcal{L}_A(\theta, g)$

Adversarial Distribution Alignment (GAN/DANN)

$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right]$$

Adversarial Support Alignment (ASA)

$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[d(g(x), \mathrm{supp}(g_\sharp q))\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[d(g(x), \mathrm{supp}(g_\sharp p))\right]$$

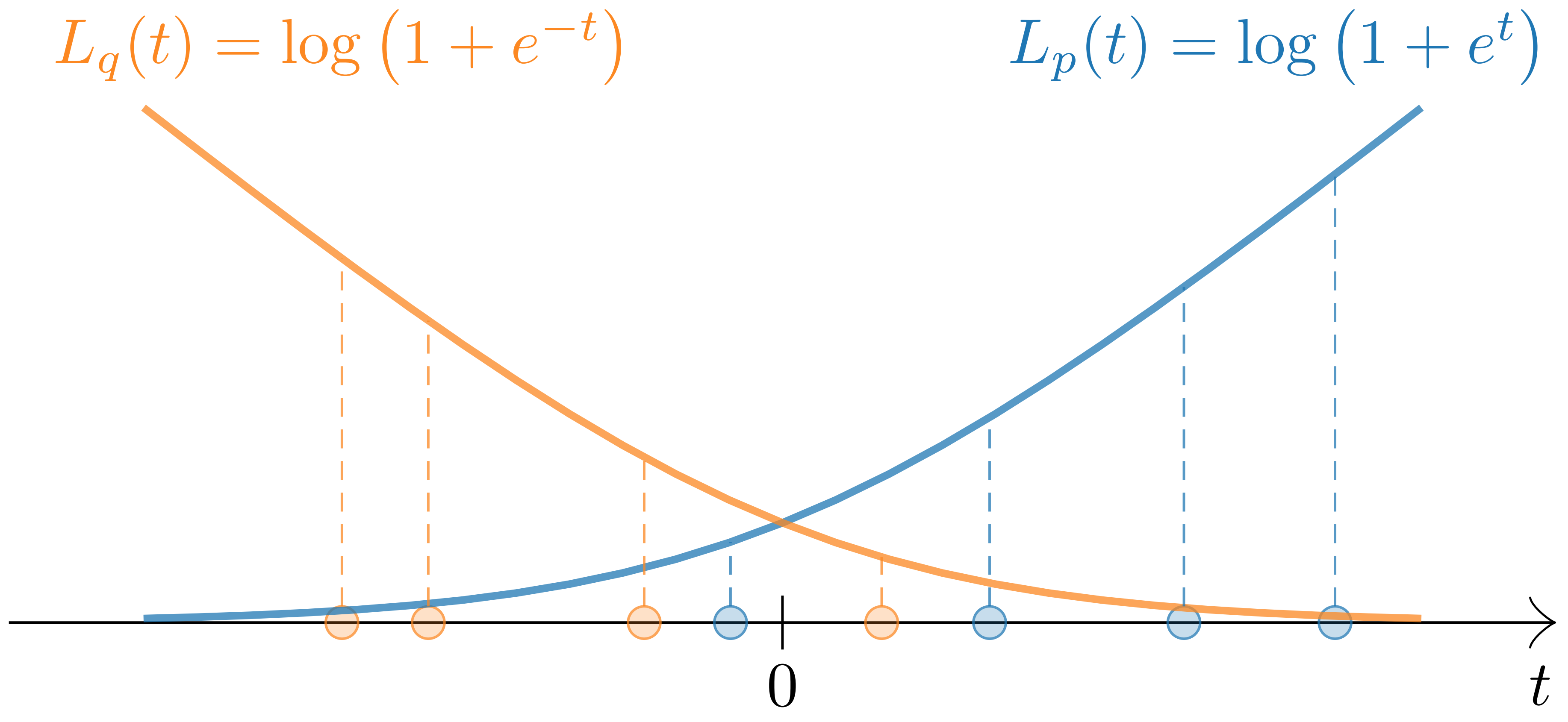
Adversarial Support Alignment

$$\mathcal{L}_D(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{g(x)}\right)\right]$$

Alignment objective: $\min_\theta \ \mathcal{L}_A(\theta, g)$

Adversarial Distribution Alignment (GAN/DANN)

$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right]$$



Adversarial Support Alignment (ASA)

$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[d(g(x), \mathrm{supp}(g_\sharp q))\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[d(g(x), \mathrm{supp}(g_\sharp p))\right]$$

Adversarial Support Alignment

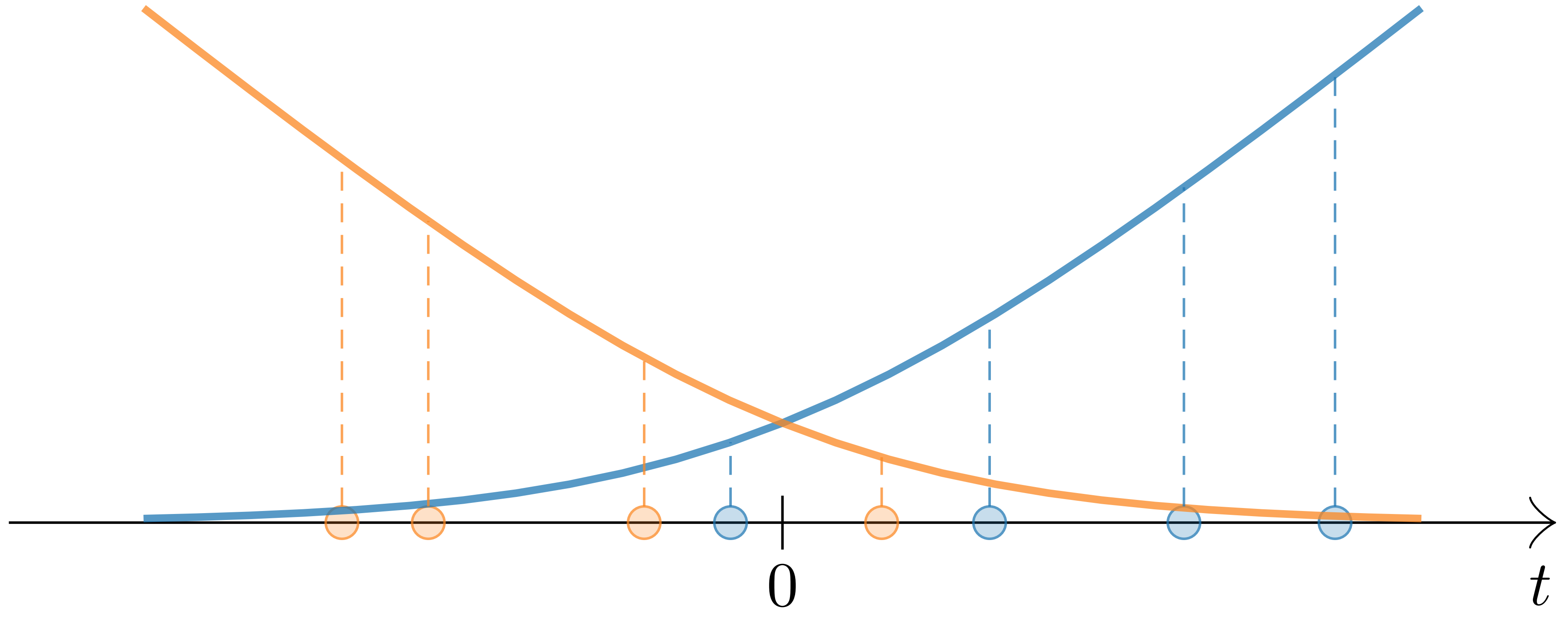
$$\mathcal{L}_D(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{g(x)}\right)\right]$$

Alignment objective: $\min\limits_{\theta} \; \mathcal{L}_A(\theta, g)$

Adversarial Distribution Alignment (GAN/DANN)

$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[\log\left(1 + e^{g(x)}\right)\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[\log\left(1 + e^{-g(x)}\right)\right]$$

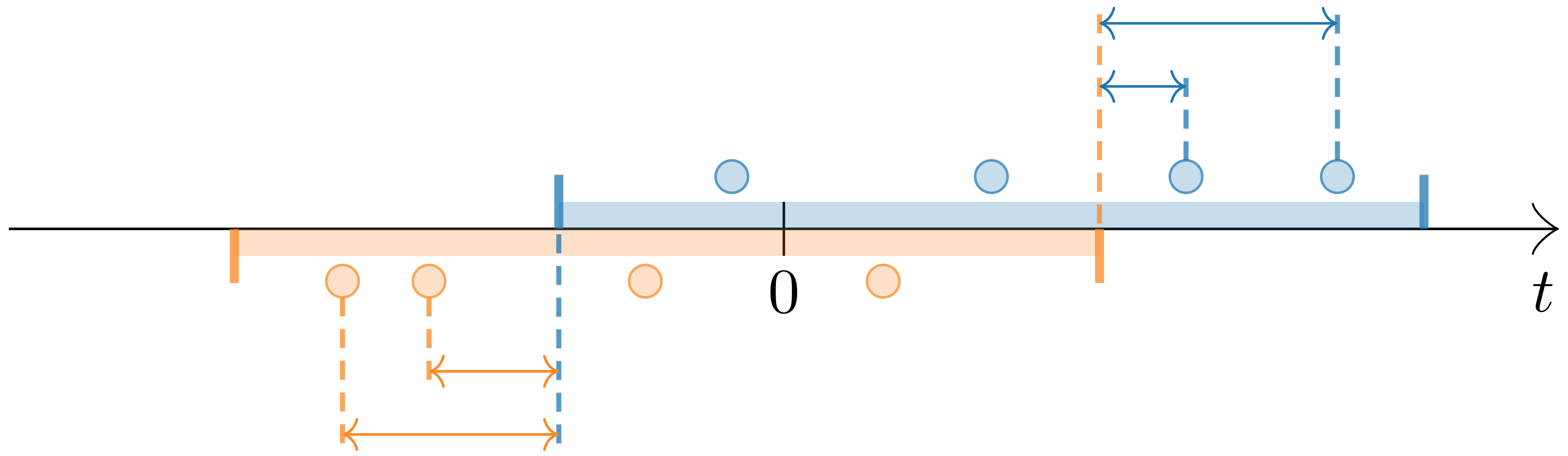
$L_q(t) = \log\left(1 + e^{-t}\right)$ $\quad$ $L_p(t) = \log\left(1 + e^{t}\right)$



Adversarial Support Alignment (ASA)

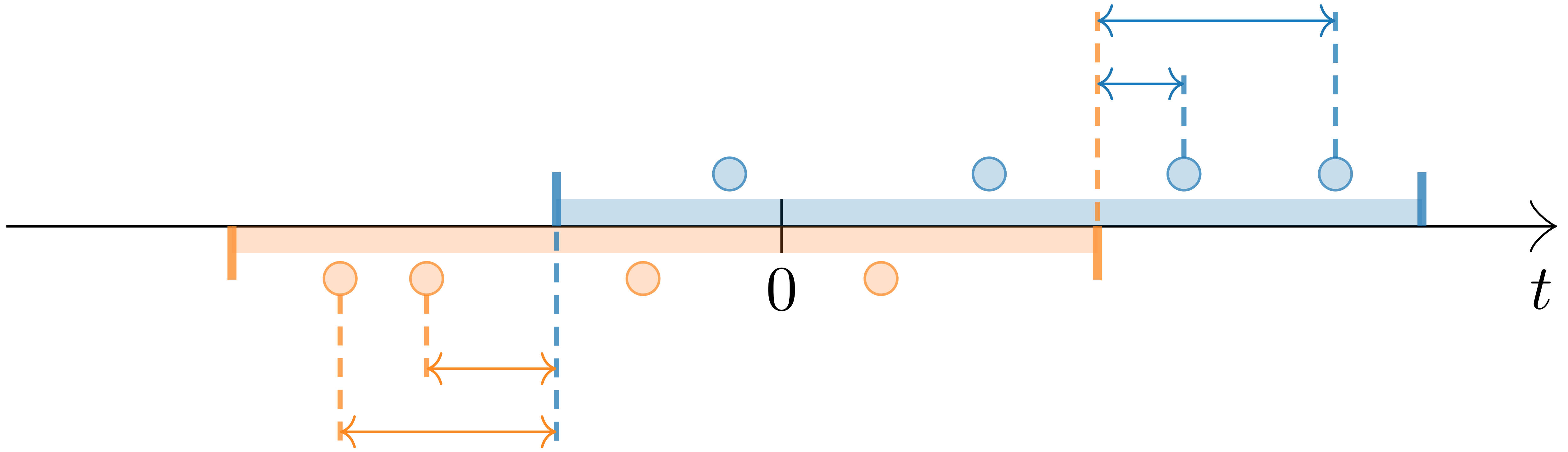
$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta}\left[d(g(x), \mathrm{supp}(g_\sharp q))\right] + \mathop{\mathbb{E}}_{x \sim q^\theta}\left[d(g(x), \mathrm{supp}(g_\sharp p))\right]$$

$L_q(t) = d(t, \mathrm{supp}(g_\sharp p))$ $\quad$ $L_p(t) = d(t, \mathrm{supp}(g_\sharp q))$



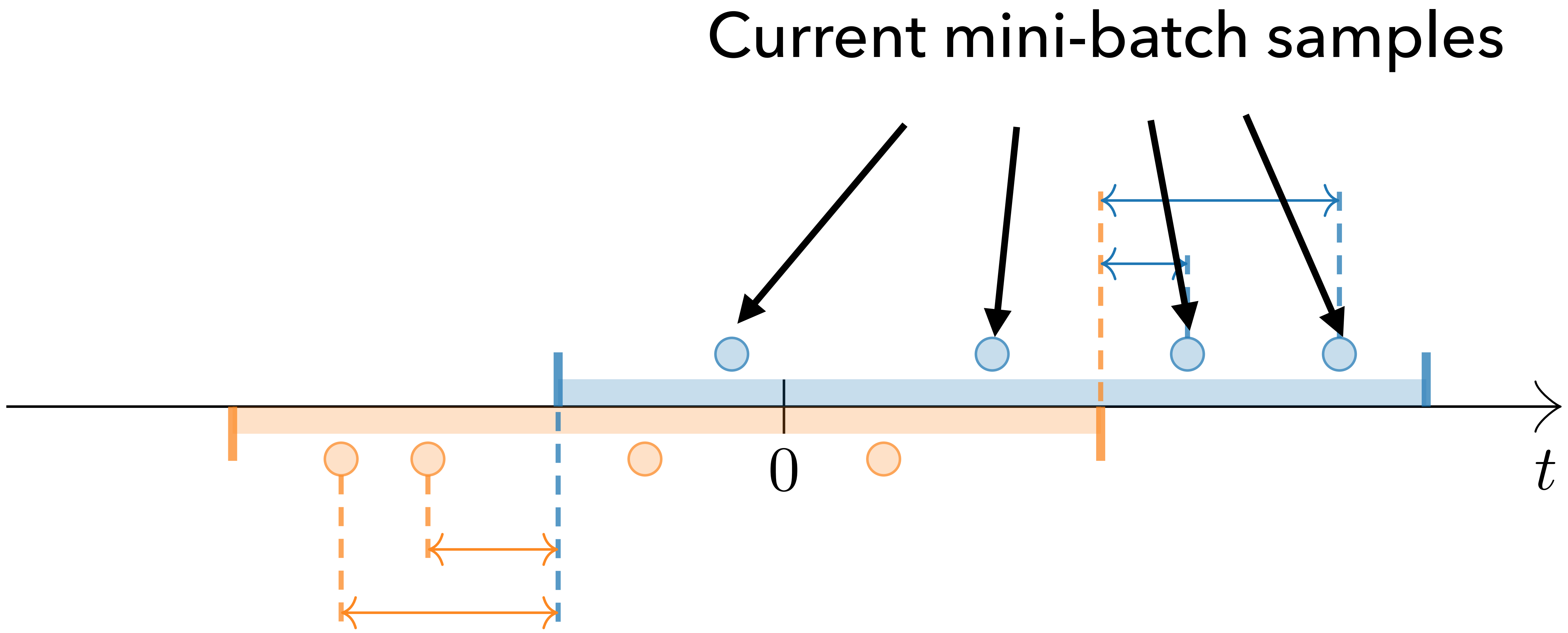
$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x\sim p^\theta}\Big[d(g(x), \mathrm{supp}(g_\sharp q))\Big] + \mathop{\mathbb{E}}_{x\sim q^\theta}\Big[d(g(x), \mathrm{supp}(g_\sharp p))\Big]$$

$$L_q(t) = d(t, \mathrm{supp}(g_\sharp p)) \qquad\qquad L_p(t) = d(t, \mathrm{supp}(g_\sharp q))$$



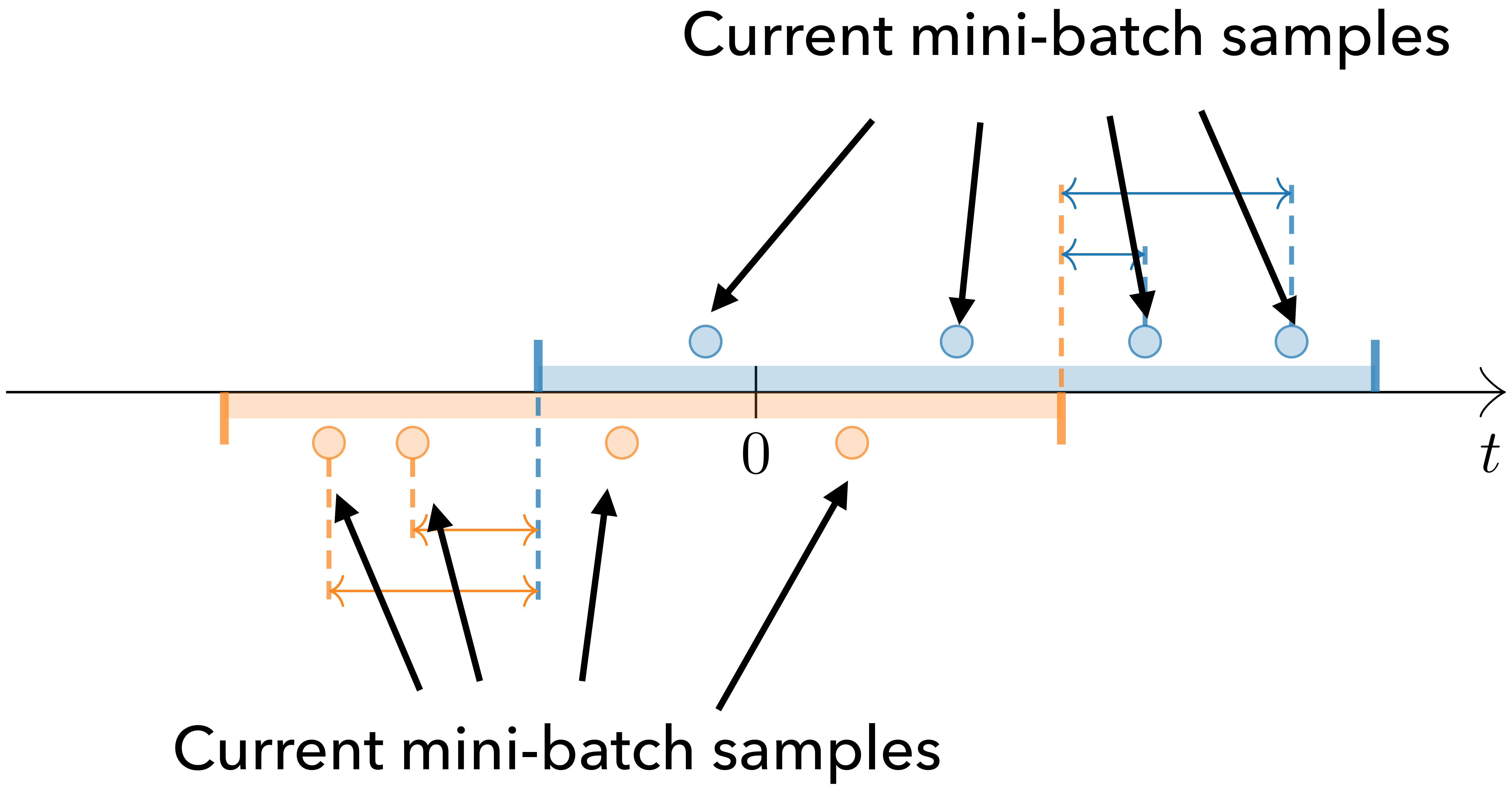
$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta} \left[d(g(x), \mathrm{supp}(g_\sharp q)) \right] + \mathop{\mathbb{E}}_{x \sim q^\theta} \left[d(g(x), \mathrm{supp}(g_\sharp p)) \right]$$

$L_q(t) = d(t, \mathrm{supp}(g_\sharp p))$ $\quad$ $L_p(t) = d(t, \mathrm{supp}(g_\sharp q))$



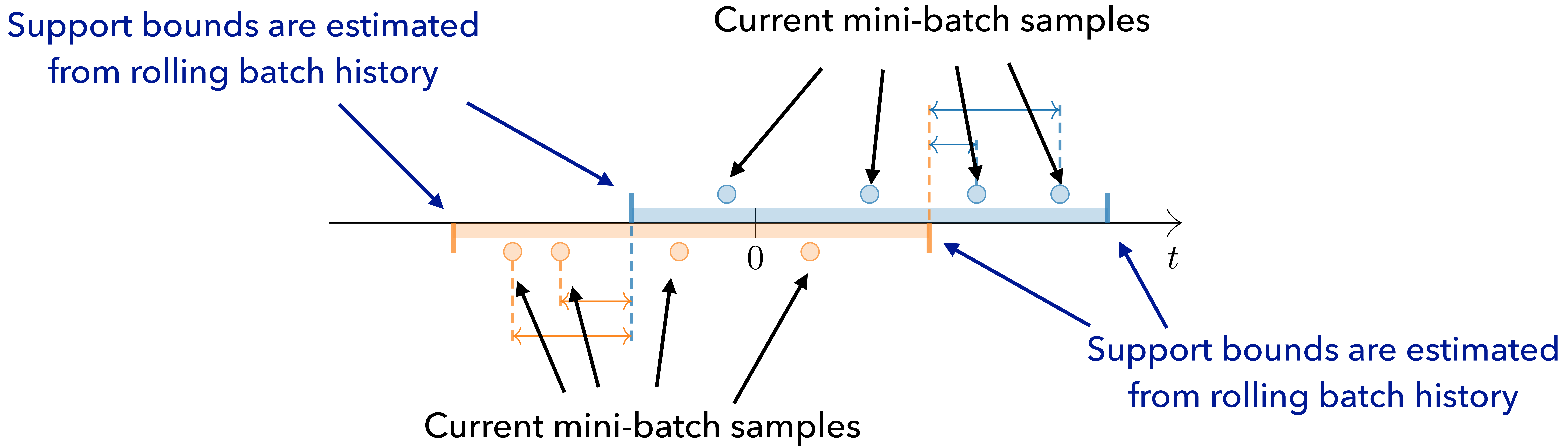
$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta} \left[d(g(x), \mathrm{supp}(g_\sharp q)) \right] + \mathop{\mathbb{E}}_{x \sim q^\theta} \left[d(g(x), \mathrm{supp}(g_\sharp p)) \right]$$

$$L_q(t) = d(t, \mathrm{supp}(g_\sharp p)) \qquad L_p(t) = d(t, \mathrm{supp}(g_\sharp q))$$



$$\mathcal{L}_A(\theta, g) = \mathop{\mathbb{E}}_{x \sim p^\theta} \left[d(g(x), \mathrm{supp}(g_\sharp q)) \right] + \mathop{\mathbb{E}}_{x \sim q^\theta} \left[d(g(x), \mathrm{supp}(g_\sharp p)) \right]$$

$$L_q(t) = d(t, \mathrm{supp}(g_\sharp p)) \qquad L_p(t) = d(t, \mathrm{supp}(g_\sharp q))$$



Results: 2D Embeddings Visualization

Toy problem: USPS→MNIST, 3 classes, 2D Embeddings

Source class distribution
[**33%**, **33%**, **33%**]

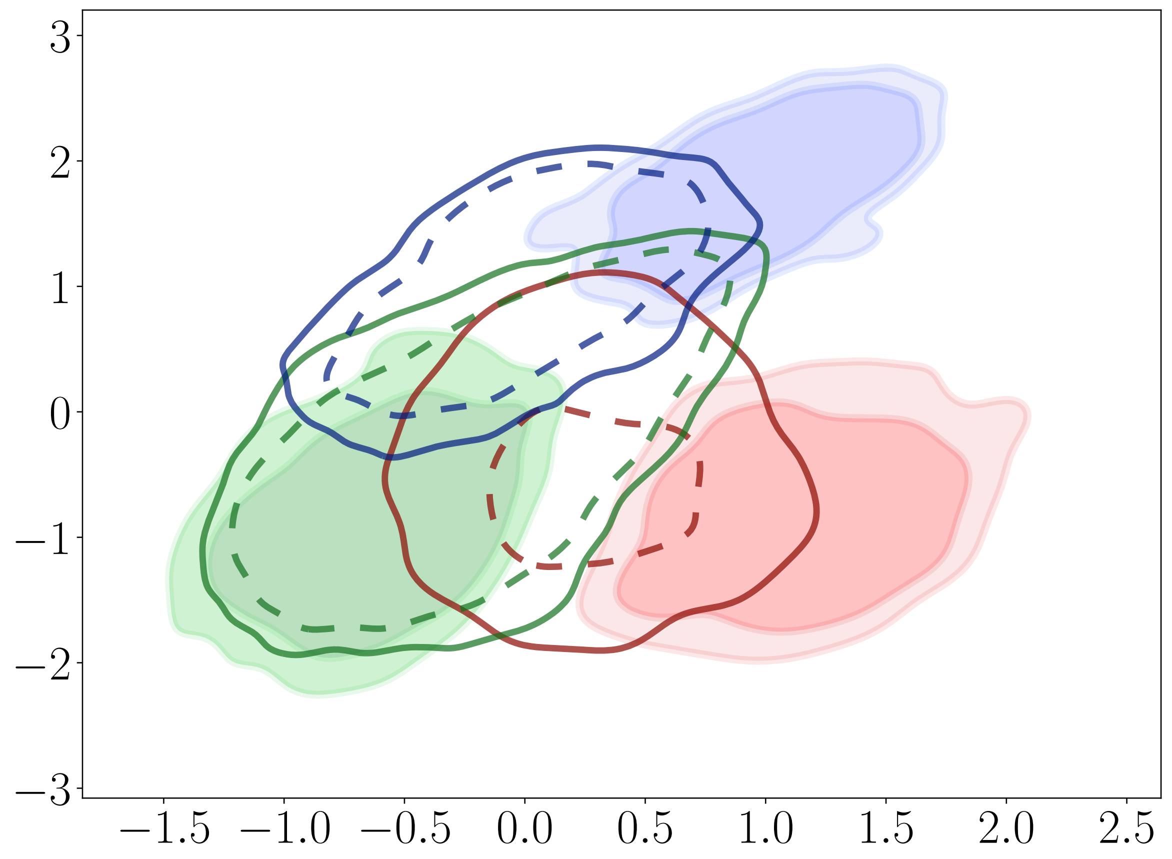
Target class distribution
[**23%**, **65%**, **12%**]

Results: 2D Embeddings Visualization

Toy problem: USPS→MNIST, 3 classes, 2D Embeddings

Source class distribution
[**33%**, **33%**, **33%**]

Target class distribution
[**23%**, **65%**, **12%**]



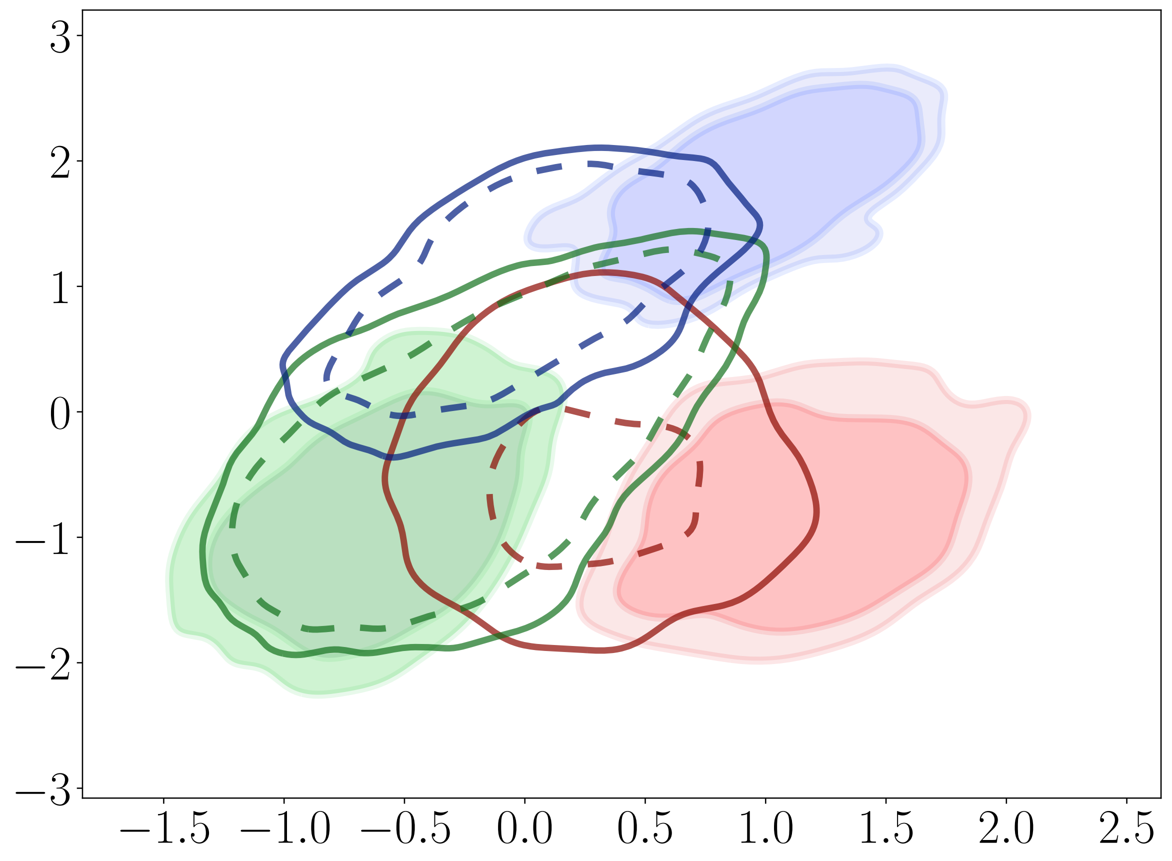
(a) No DA (avg acc: 63%)
$\mathcal{D}_W(p_Z^\theta, q_Z^\theta) = 0.78$
$\mathcal{D}_\triangle(p_Z^\theta, q_Z^\theta) = 0.10$

Results: 2D Embeddings Visualization

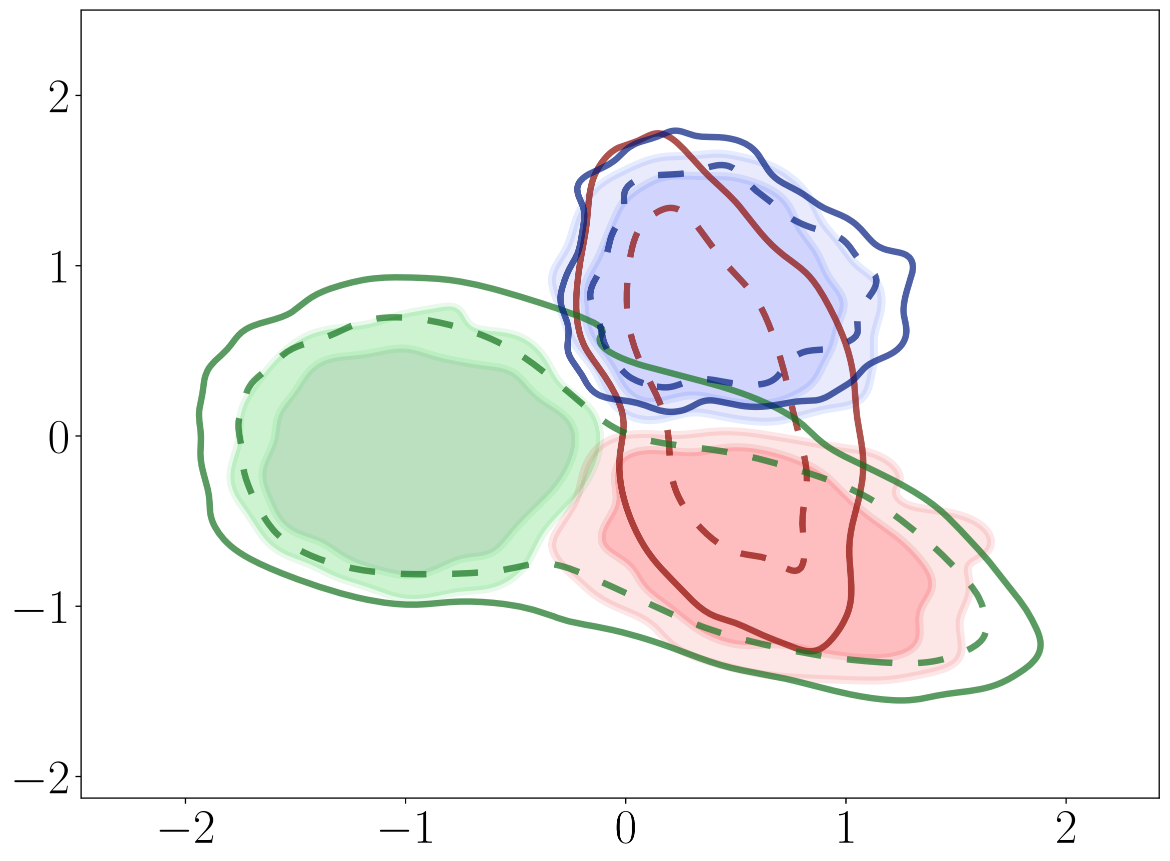
Toy problem: USPS→MNIST, 3 classes, 2D Embeddings

Source class distribution
[**33%**, **33%**, **33%**]

Target class distribution
[**23%**, **65%**, **12%**]



(a) No DA (avg acc: 63%)
$\mathcal{D}_W(p_Z^\theta, q_Z^\theta) = 0.78$
$\mathcal{D}_\triangle(p_Z^\theta, q_Z^\theta) = 0.10$



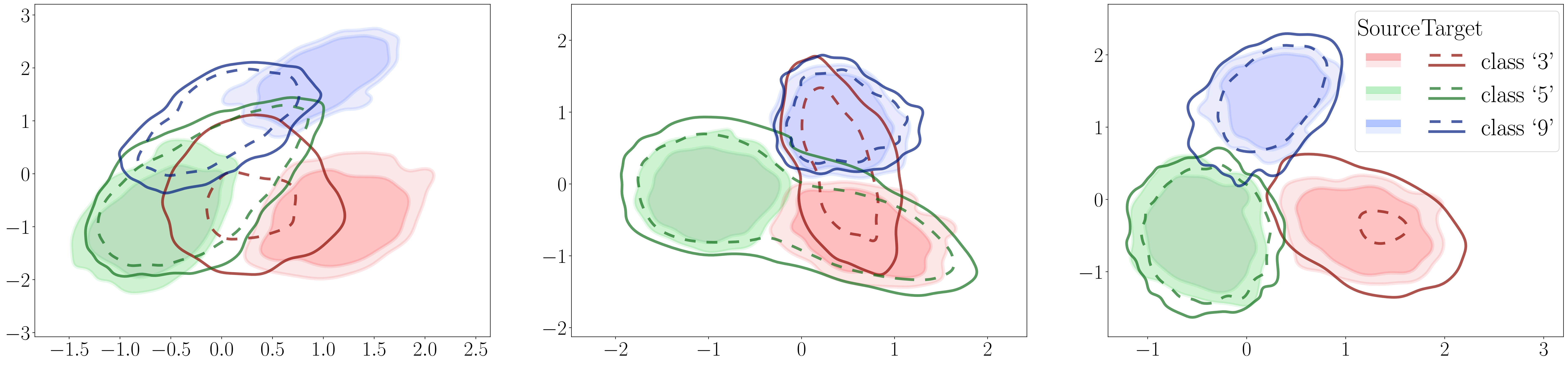
(b) DANN (avg acc: 75%)
$\mathcal{D}_W(p_Z^\theta, q_Z^\theta) = 0.07$
$\mathcal{D}_\triangle(p_Z^\theta, q_Z^\theta) = 0.02$

Results: 2D Embeddings Visualization

Toy problem: USPS→MNIST, 3 classes, 2D Embeddings

Source class distribution
[33%, 33%, 33%]

Target class distribution
[23%, 65%, 12%]



(a) No DA (avg acc: 63%)
$\mathcal{D}_W(p_Z^\theta, q_Z^\theta) = 0.78$
$\mathcal{D}_\triangle(p_Z^\theta, q_Z^\theta) = 0.10$

(b) DANN (avg acc: 75%)
$\mathcal{D}_W(p_Z^\theta, q_Z^\theta) = 0.07$
$\mathcal{D}_\triangle(p_Z^\theta, q_Z^\theta) = 0.02$

(c) ASA-abs (avg acc: 94%)
$\mathcal{D}_W(p_Z^\theta, q_Z^\theta) = 0.59$
$\mathcal{D}_\triangle(p_Z^\theta, q_Z^\theta) = 0.03$

Results: USPS → MNIST, SmallCNN

Average and minimum class accuracy (%) on USPS→MNIST across different levels of shifts in label distributions (α).

Group	Algorithm	$\alpha = 0.0$ no shift		$\alpha = 1.0$		$\alpha = 1.5$		$\alpha = 2.0$ severe shift	
		average	min	average	min	average	min	average	min
	No DA	71.9	20.3	72.9	25.8	71.3	27.5	71.3	16.6
Distribution Alignment	DANN	97.8	96.0	83.5	25.1	70.0	01.1	57.8	00.9
	VADA	**98.0**	96.2	88.2	48.9	78.2	06.6	61.9	01.4
Relaxed Distribution Alignment	IWDAN	97.5	95.7	95.7	81.3	86.5	15.2	74.4	07.3
	IWCDAN	**98.0**	**96.6**	**96.7**	85.1	91.3	66.5	77.5	22.2
	sDANN-4	87.4	05.6	94.9	**85.7**	86.8	21.6	81.5	39.3
Support Alignment (ours)	ASA-sq	93.7	89.2	92.3	83.5	90.9	69.9	87.2	62.5
	ASA-abs	94.1	88.9	92.8	78.9	**92.5**	**82.4**	**90.4**	**68.4**

~0% worst-class accuracy under severe shift

Our method is robust to distribution shifts

Results: Larger Datasets

STL10 → CIFAR10
DeepCNN

Algorithm	$\alpha = 0.0$ no shift: average	$\alpha = 0.0$ no shift: min	$\alpha = 2.0$ severe shift: average	$\alpha = 2.0$ severe shift: min
No DA	69.9	49.8	65.8	43.7
DANN	75.3	54.6	63.3	27.0
VADA	**76.7**	**56.9**	63.2	25.5
IWDAN	69.9	50.5	64.4	36.8
IWCDAN	70.1	47.8	64.5	37.0
sDANN-4	71.8	52.1	66.4	39.0
ASA-sq	71.7	52.9	**68.1**	**44.7**
ASA-abs	71.6	49.0	67.8	40.9

VisDA-17
ResNet50

Algorithm	$\alpha = 0.0$ no shift: average	$\alpha = 0.0$ no shift: min	$\alpha = 2.0$ severe shift: average	$\alpha = 2.0$ severe shift: min
No DA	49.5	22.2	45.3	19.5
DANN	**75.4**	36.7	43.1	03.6
VADA	75.3	40.5	43.9	08.5
IWDAN	73.2	31.7	45.1	04.6
IWCDAN	71.6	27.6	38.3	00.6
sDANN-4	72.4	37.8	50.7	18.6
ASA-sq	64.9	35.7	51.9	18.3
ASA-abs	64.8	**40.6**	**52.5**	**19.7**

Adversarial Support Alignment: Summary

Support alignment: novel training criterion, an alternative to distribution alignment

- **Support divergence** defined for continuous distributions
  - **Spectrum of alignment criteria** within **relaxed OT** framework
  - Distribution alignment / relaxed distribution alignment / support alignment
- Analysis of support discrepancy in the discriminator output space
  - Log-loss discriminator **preserves support discrepancy**
  - Not all discriminators have this property (e.g. linear discriminators, Wasserstein discriminators)
- Practical method: **log-loss discriminator + support difference + history buffers**

Experimental validation: unsupervised domain adaptation under label distribution shift

- Image classification UDA benchmarks, USPS-MNIST, CIFAR-STL, VisDA17
- Robust performance in the face of label distribution shift, **improved worst class accurac**y

Chapter IV
Compositional Sculpting of Iterative Generative Processes

Composition of Generative Models

Large-scale general-purpose pre-training is becoming ubiquitous

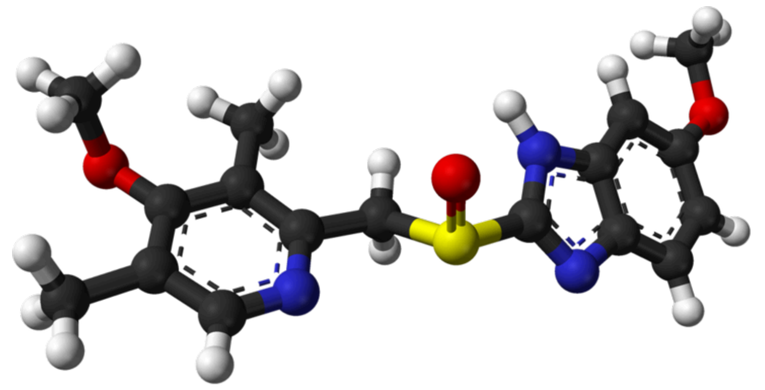
- Need to re-use and adapt **pre-trained models** for **new tasks**

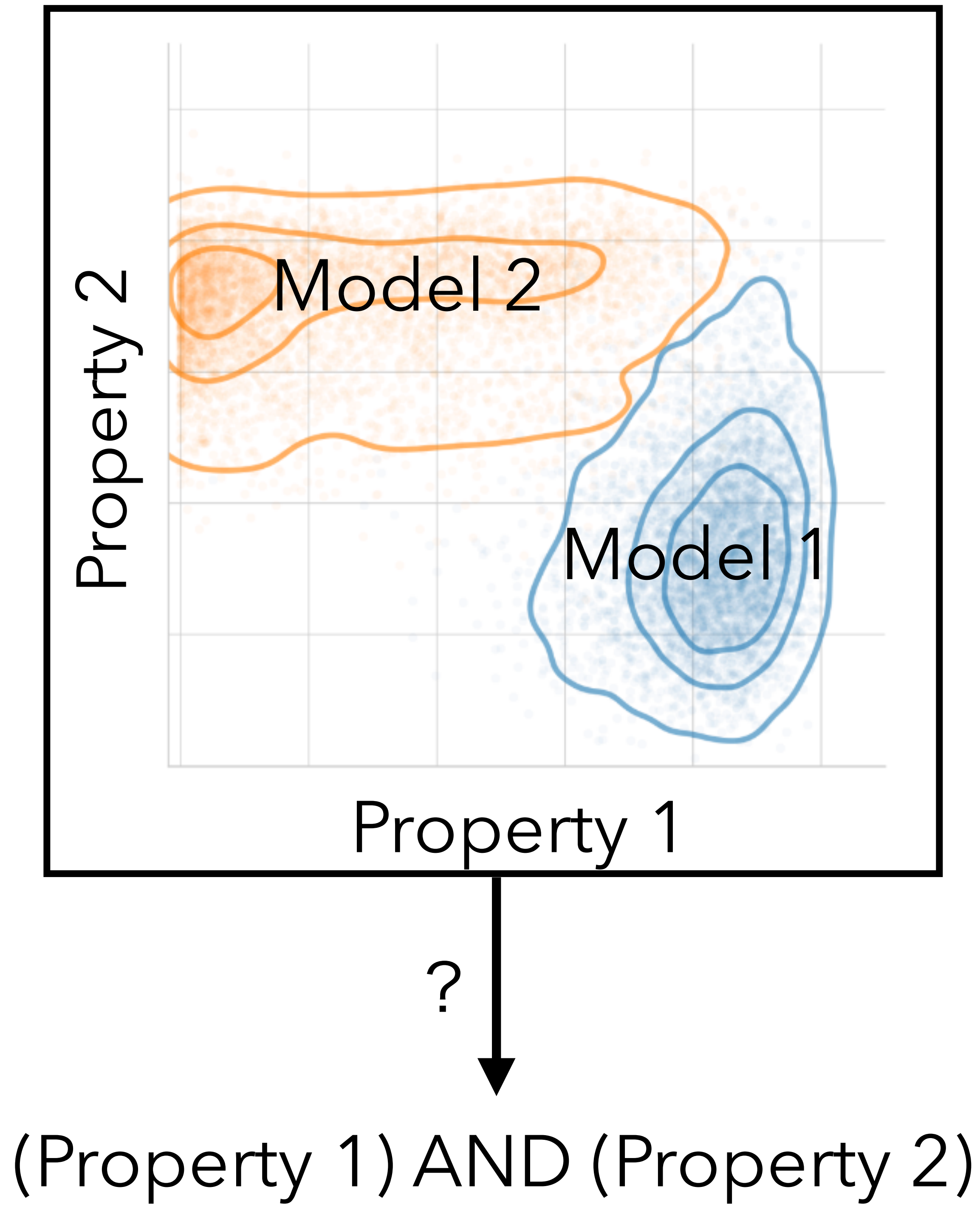
Applications: multi-objective generation (e.g. drug-like molecules)

- Need to combine knowledge from multiple sources (models, datasets)
- Need to explore trade-offs between multiple criteria

Composition is a powerful modeling tool

- Increases model capacity
- Enables control of sampling distributions



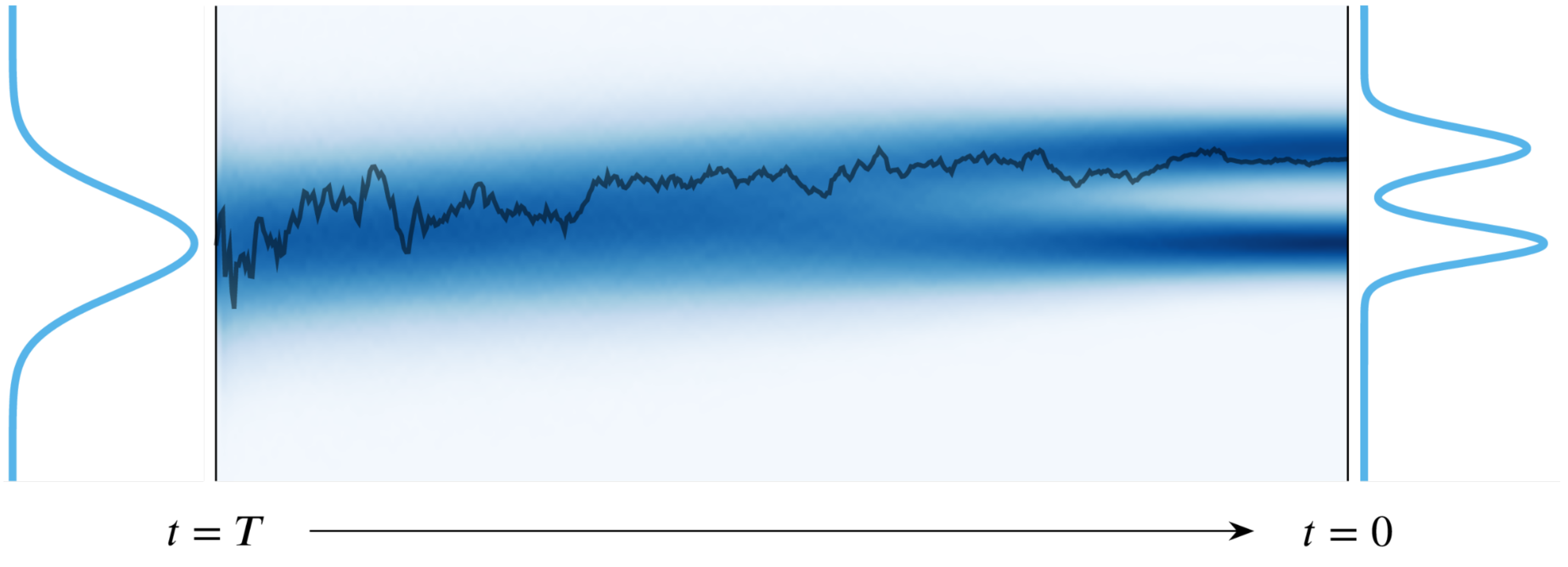


Iterative Generative Processes

Diffusion model generates trajectories $\tau = \left(x_T \to \{x_t\}_{t=0}^{T} \to x_0\right)$ with terminal state distribution $p(x_0)$ by following a backward SDE

$$dx_t = \left[f_t(x_t) - g_t^2 \underbrace{s_t(x_t;\theta)}_{\approx \nabla_{x_t} \log p_t(x_t)}\right] dt + g_t \, d\overline{w}_t,$$

corresponding to a forward noising process $dx_t = f_t(x_t)\, dt + g_t \, dw_t$

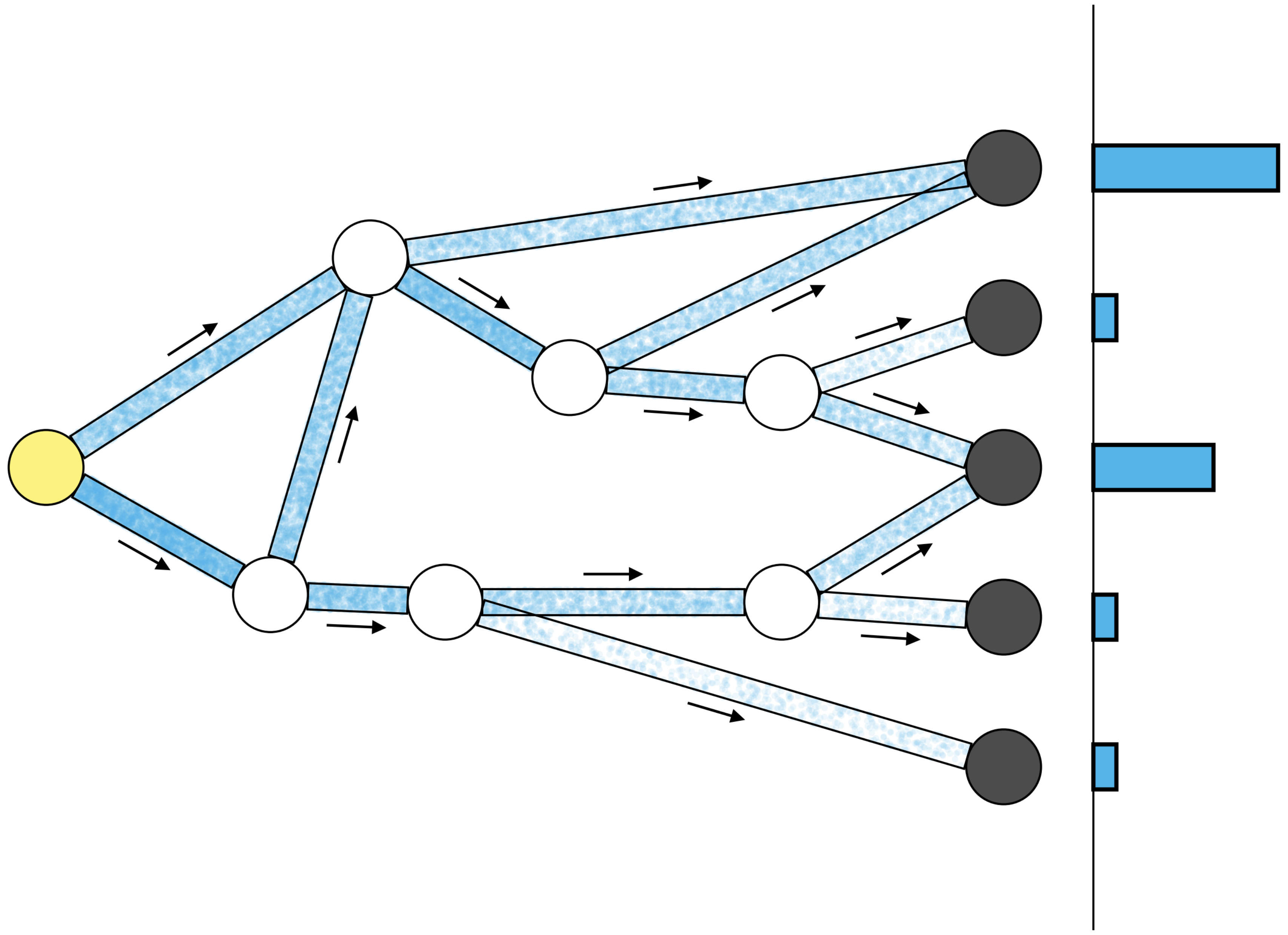


"Score-Based Generative Modeling through Stochastic Differential Equations"

[Song et al, ICLR 2021]

GFlowNet generates trajectories $\tau = \left(s_0 \to \ldots \to s_{n-1} \to x\right)$ with terminal distribution $p(x) = \frac{R(x)}{Z}$ by following a forward policy

$$p_F(s_t \mid s_{t-1}; \theta)$$



"Flow Network based Generative Models for Non-Iterative Diverse Candidate Generation"

[Bengio et al, NeurIPS 2021]

	Models	Composition Operations	Sampling Algorithm
[Hinton, Neural Computation 2002] [Du et al, NeurIPS 2020]	Energy-based models (EBMs) $p_i(x) \propto \exp(-E_i(x;\theta))$	Principle: energy-function arithmetic Product: $\frac{1}{Z}\, p_1(x)\, p_2(x)$ Negation: $\frac{1}{Z}\, \frac{p_1(x)}{\left(p_2(x)\right)^{\gamma}}$	MCMC Langevin dynamics

Base models: $$p_1(x) = \frac{1}{Z_1} \exp\left\{-E_1(x)\right\}, \quad p_2(x) = \frac{1}{Z_2} \exp\left\{-E_2(x)\right\}$$

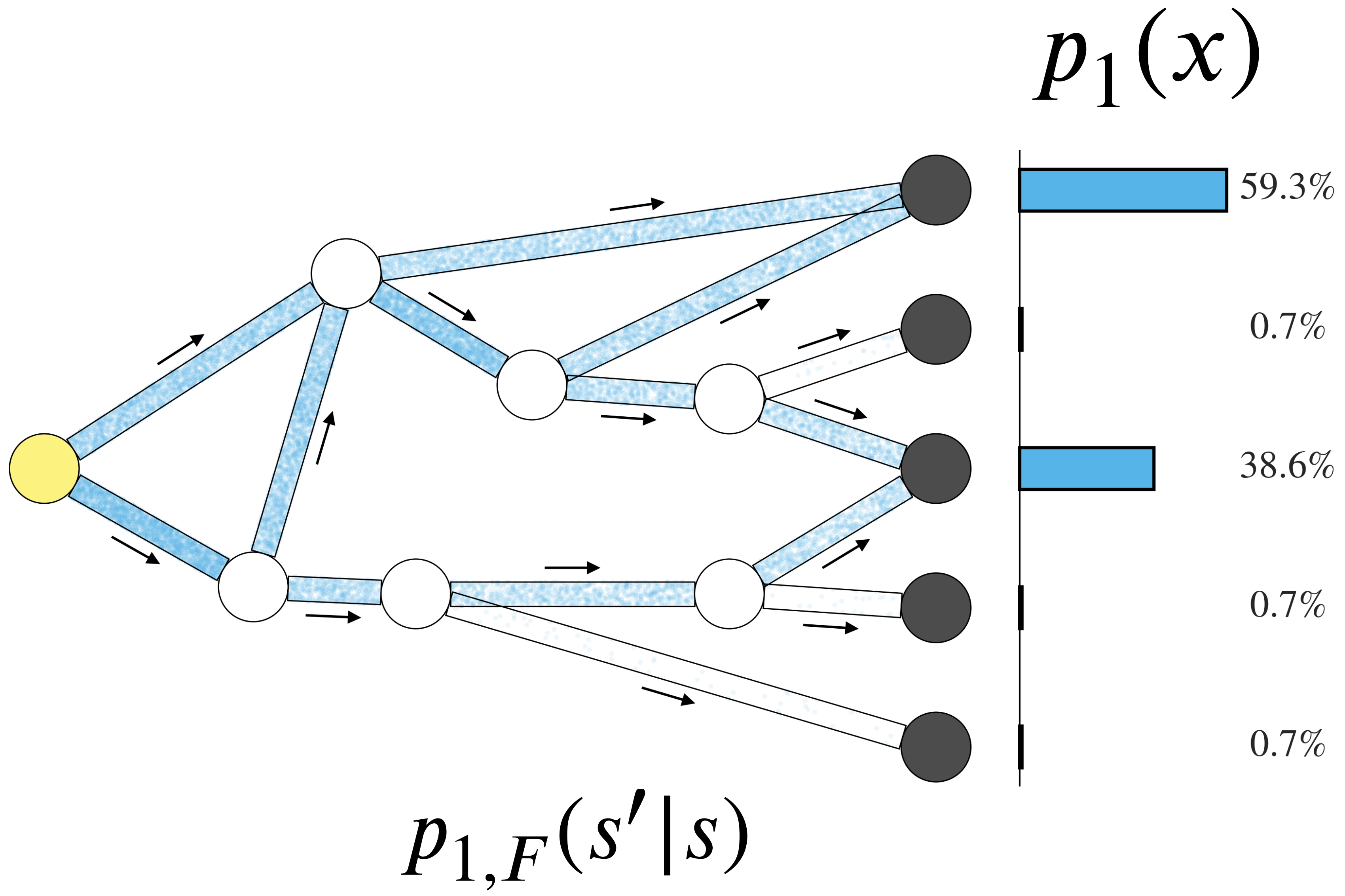
Product: $$p_{\text{prod}}(x) \propto p_1(x) p_2(x) \propto \exp\left\{-\left(E_1(x) + E_2(x)\right)\right\}$$

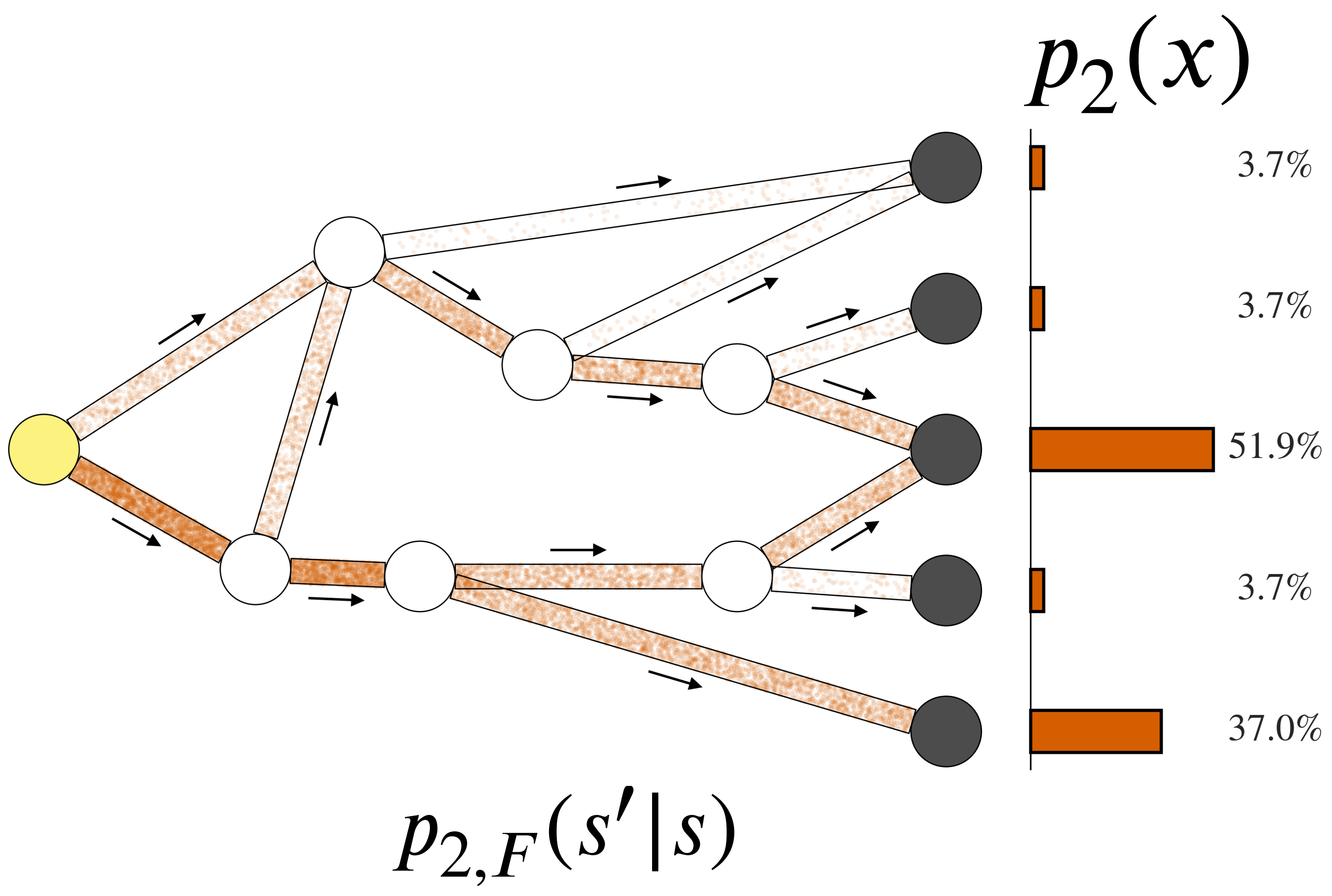
Negation: $$p_{\text{neg}}(x) \propto \frac{p_1(x)}{\left(p_2(x)\right)^{\gamma}} \propto \exp\left\{-\left(E_1(x) - \gamma E_2(x)\right)\right\}$$

	Models	Composition Operations	Sampling Algorithm
[Hinton, Neural Computation 2002] [Du et al, NeurIPS 2020]	Energy-based models (EBMs) $p_i(x) \propto \exp(-E_i(x;\theta))$	Principle: energy-function arithmetic Product: $\frac{1}{Z} p_1(x)\, p_2(x)$ Negation: $\frac{1}{Z} \frac{p_1(x)}{\left(p_2(x)\right)^\gamma}$	MCMC Langevin dynamics
[Liu et al, ECCV 2022] [Du et al, ICML 2023]	Diffusion models $p_i(x):\ s_{i,t}(x_t;\theta) \approx \nabla_{x_t} \log p_{i,t}(x_t)$	Principle: score-function arithmetic Product: $\frac{1}{Z} p_1(x)\, p_2(x)$ Negation: $\frac{1}{Z} \frac{p_1(x)}{\left(p_2(x)\right)^\gamma}$	Diffusion sampling + annealed MCMC

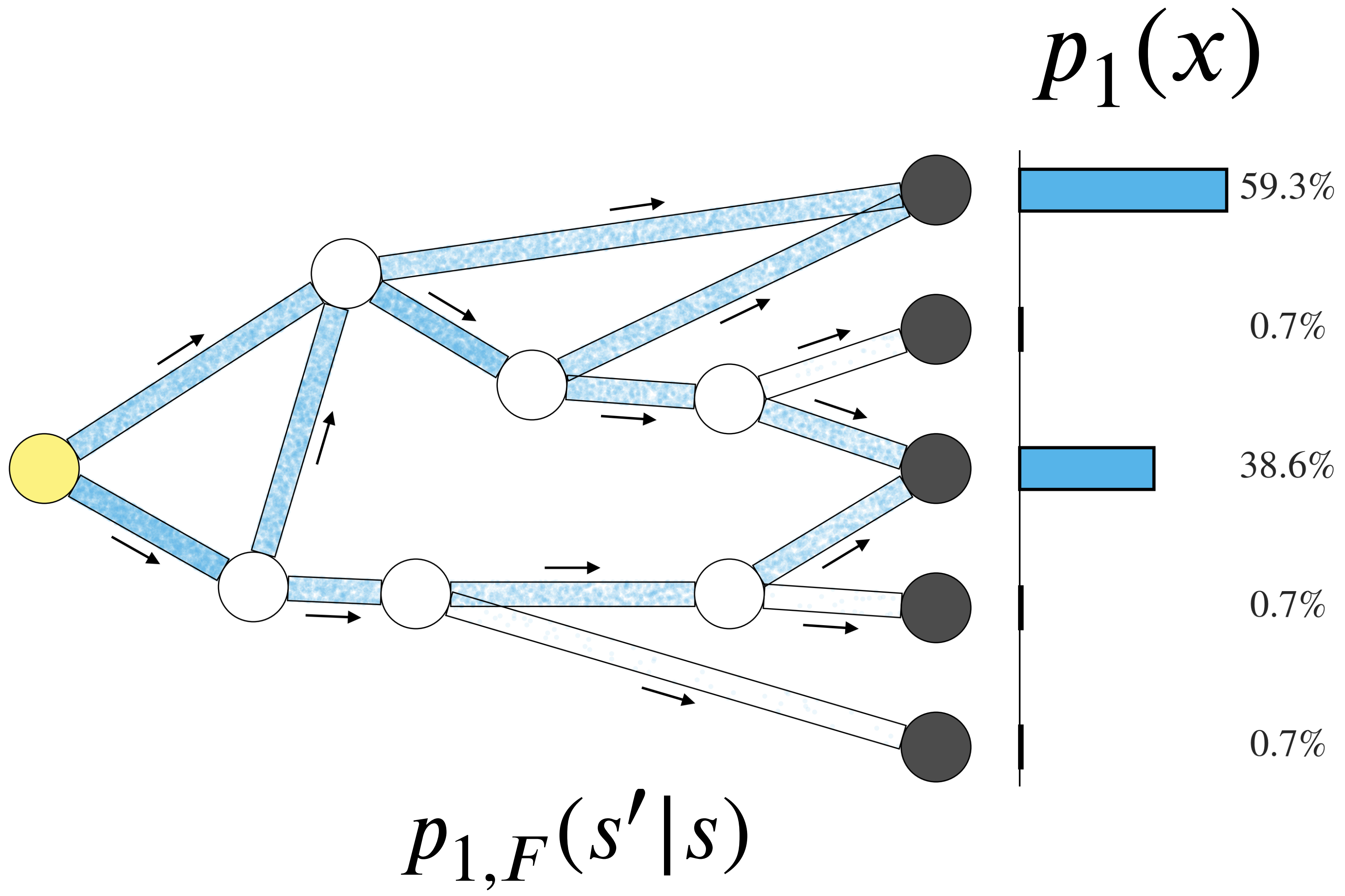
Challenge: iterative generative processes (Diffusion models & GFlowNets) impose delicate balance conditions

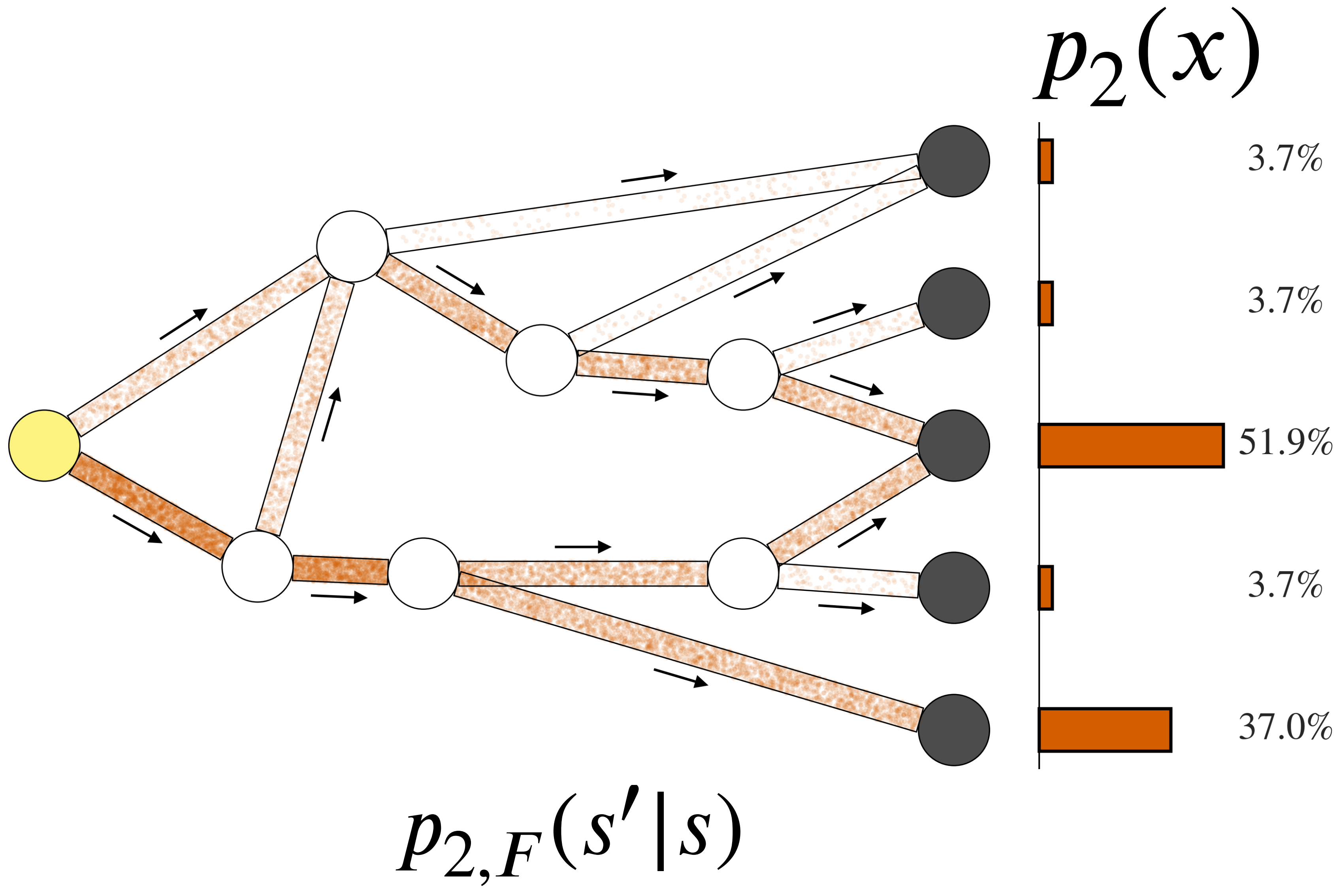
Composition Tools: Mixtures





Composition Tools: Mixtures



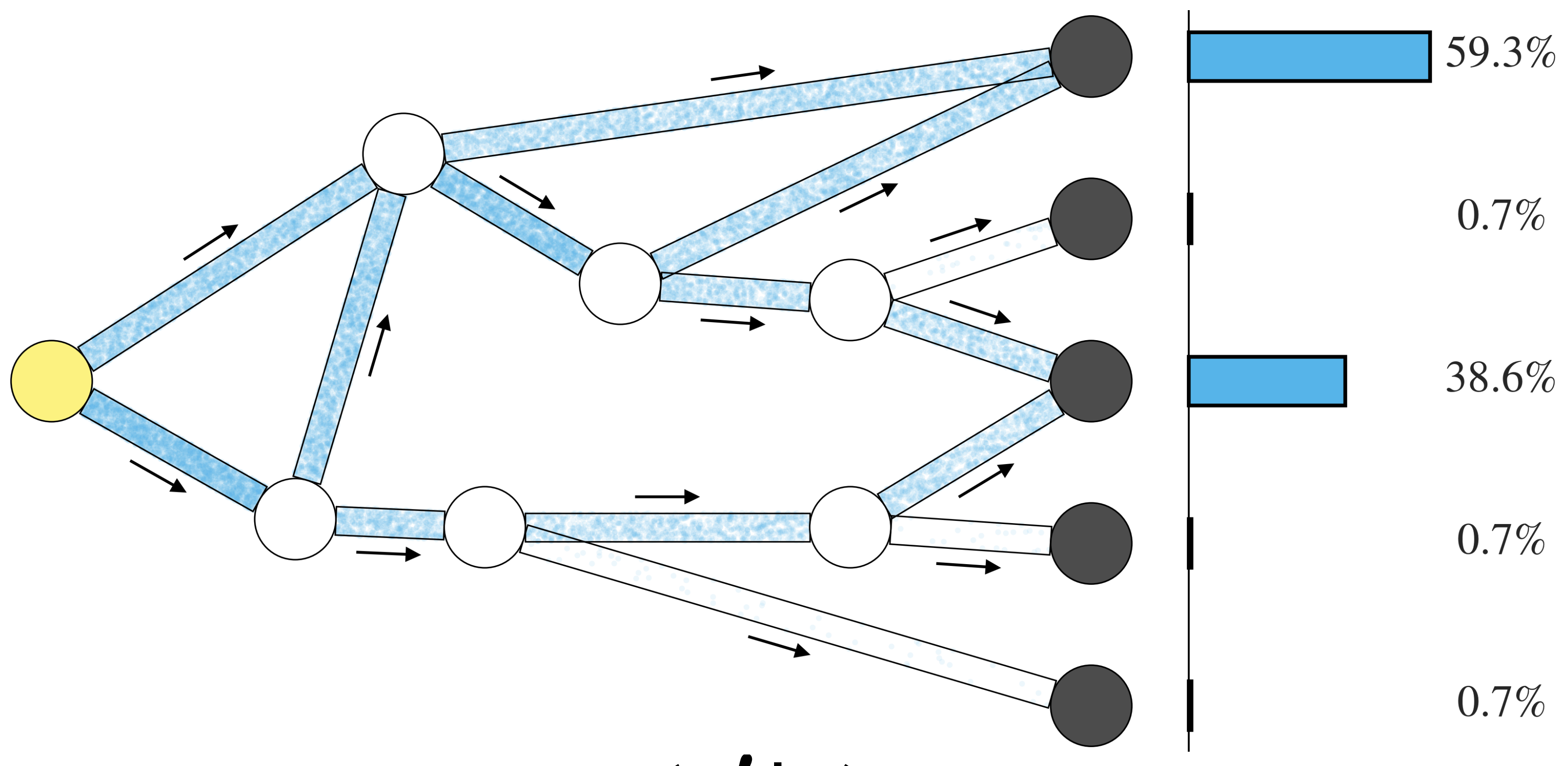


Mixture

$$p_M(x) = \frac{1}{2}p_1(x) + \frac{1}{2}p_2(x)$$

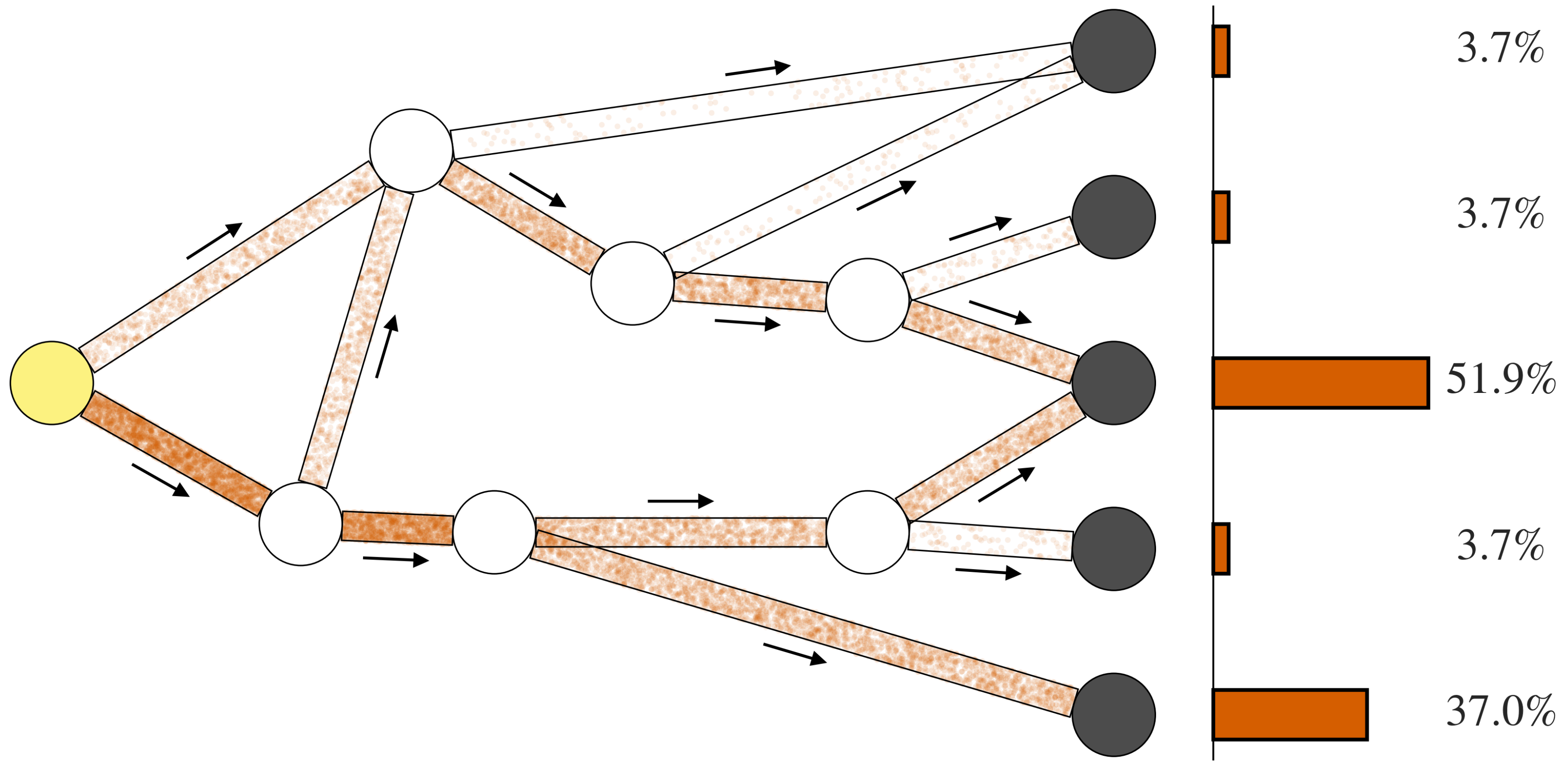
Composition Tools: Mixtures

$p_1(x)$



$p_{1,F}(s'|s)$

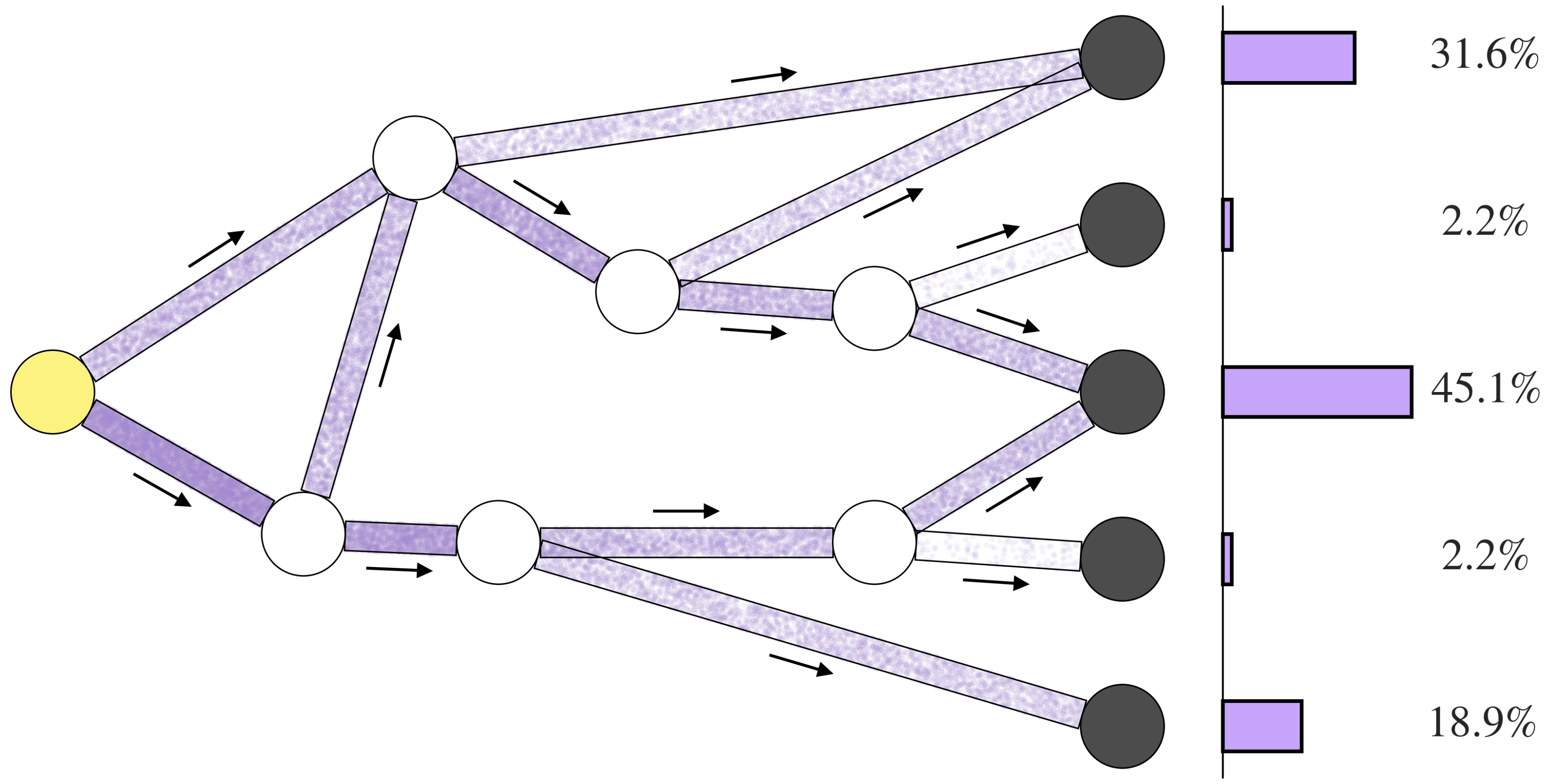
$p_2(x)$



$p_{2,F}(s'|s)$

Mixture

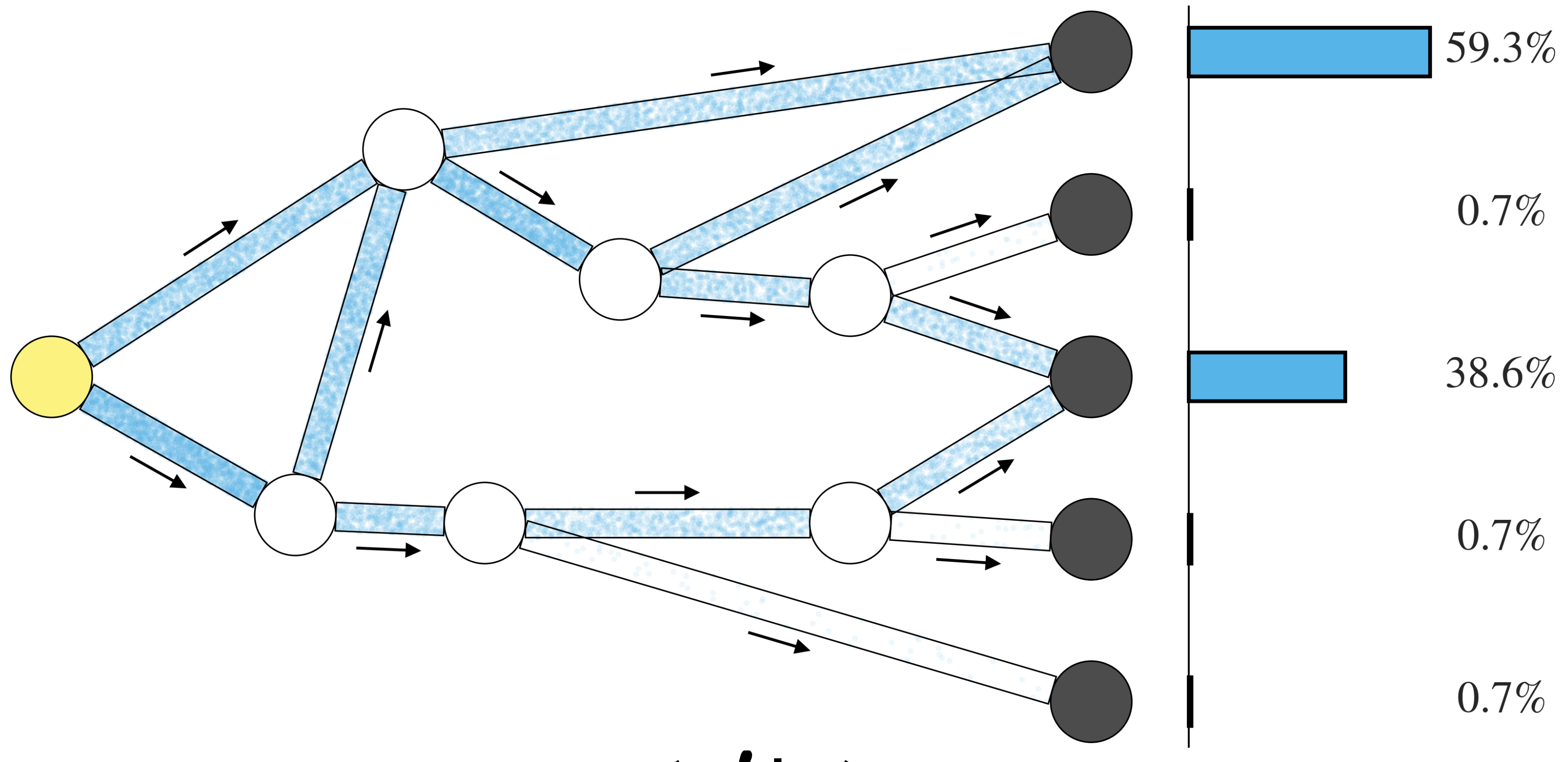
$$p_M(x) = \frac{1}{2}p_1(x) + \frac{1}{2}p_2(x)$$



$$p_{M,F}(s'|s) = \sum_{i=1}^{2} p(y=i|s)p_{i,F}(s'|s)$$

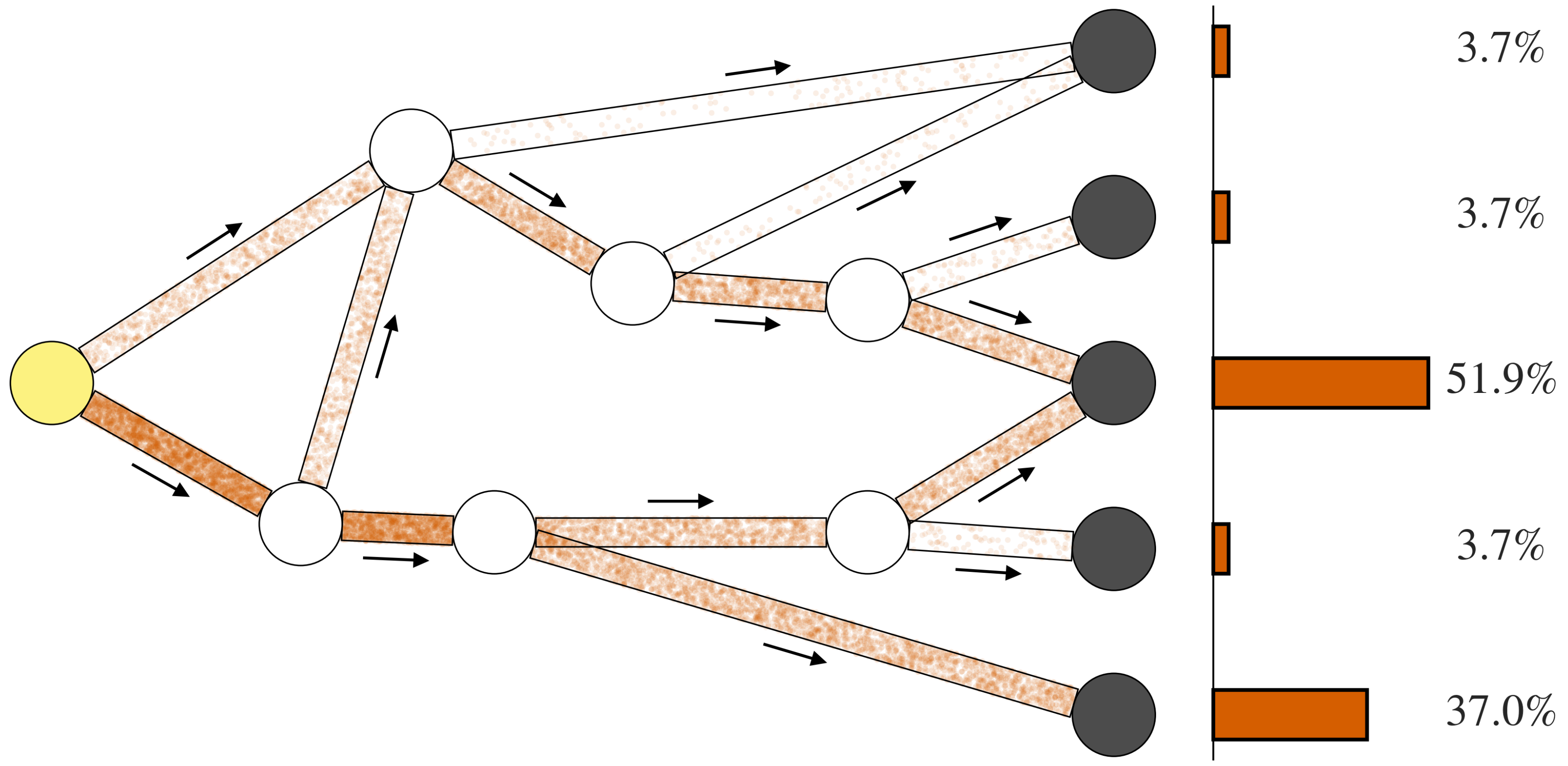
Composition Tools: Mixtures

$p_1(x)$



$p_{1,F}(s'|s)$

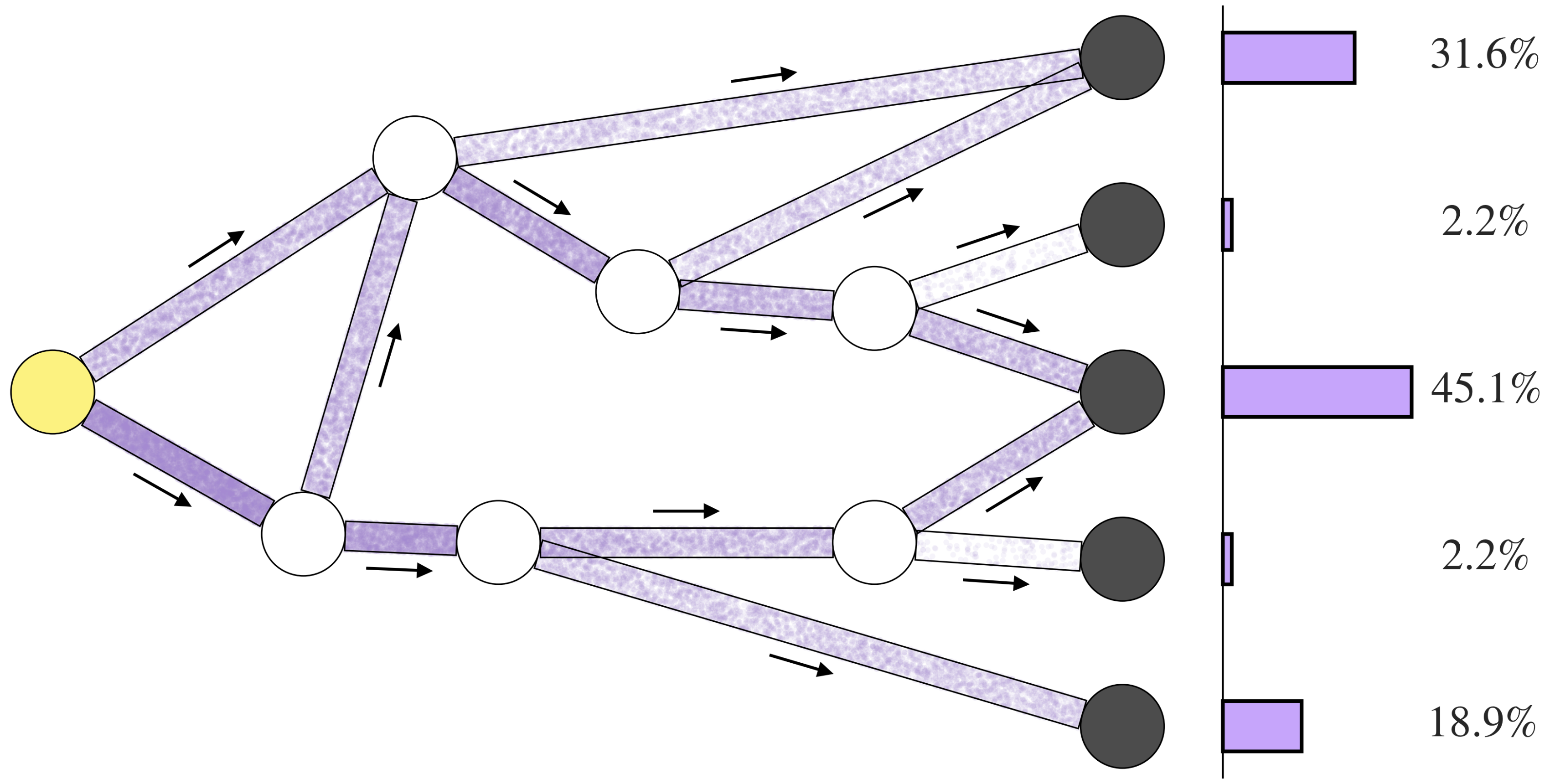
$p_2(x)$



$p_{2,F}(s'|s)$

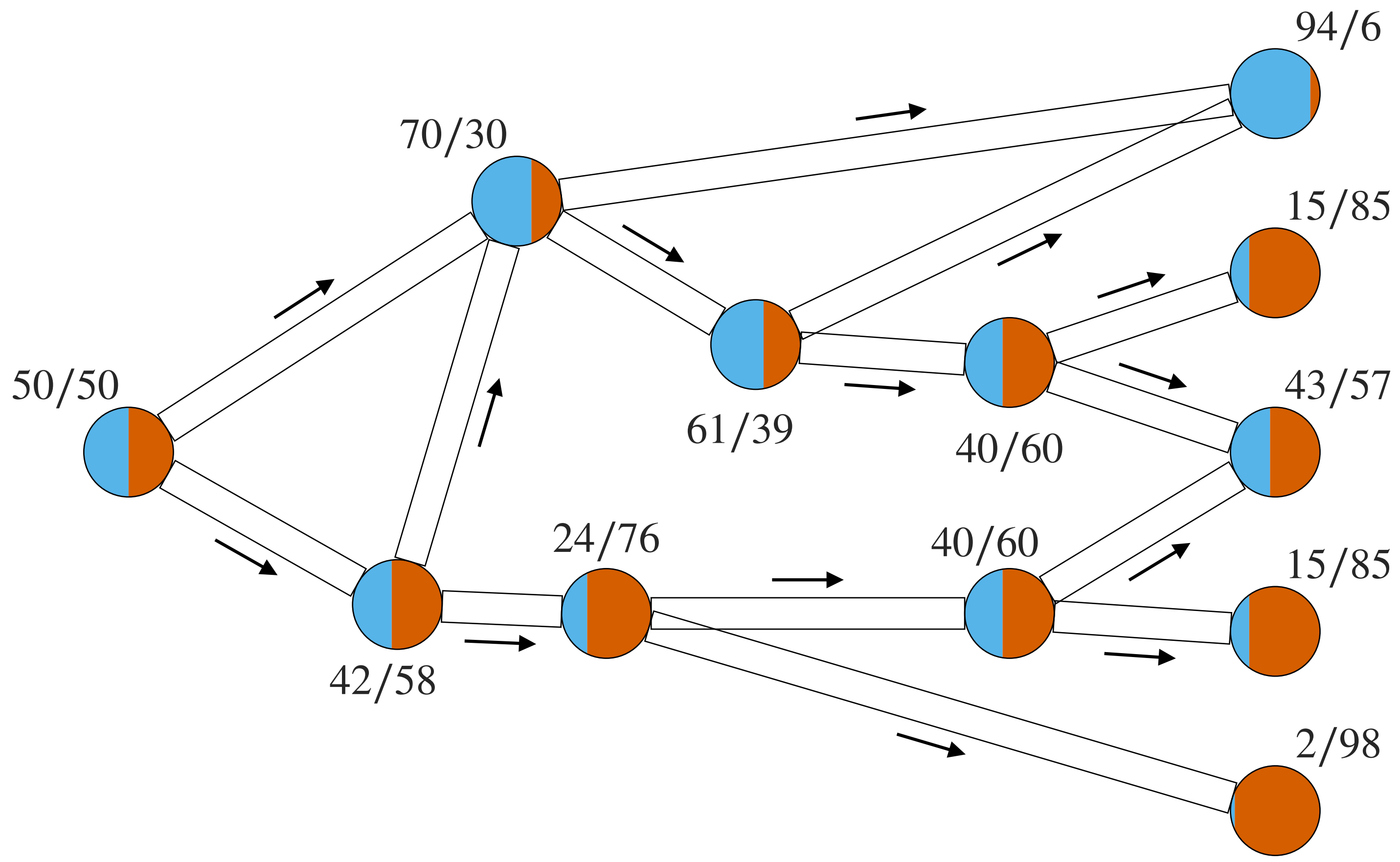
Mixture

$$p_M(x) = \frac{1}{2}p_1(x) + \frac{1}{2}p_2(x)$$



$$p_{M,F}(s'|s) = \sum_{i=1}^{2} p(y=i|s)p_{i,F}(s'|s)$$

Classifier



$$p(y=i|s) = \frac{p_i(s)}{p_1(s) + p_2(s)}$$

Composition Tools: Mixtures

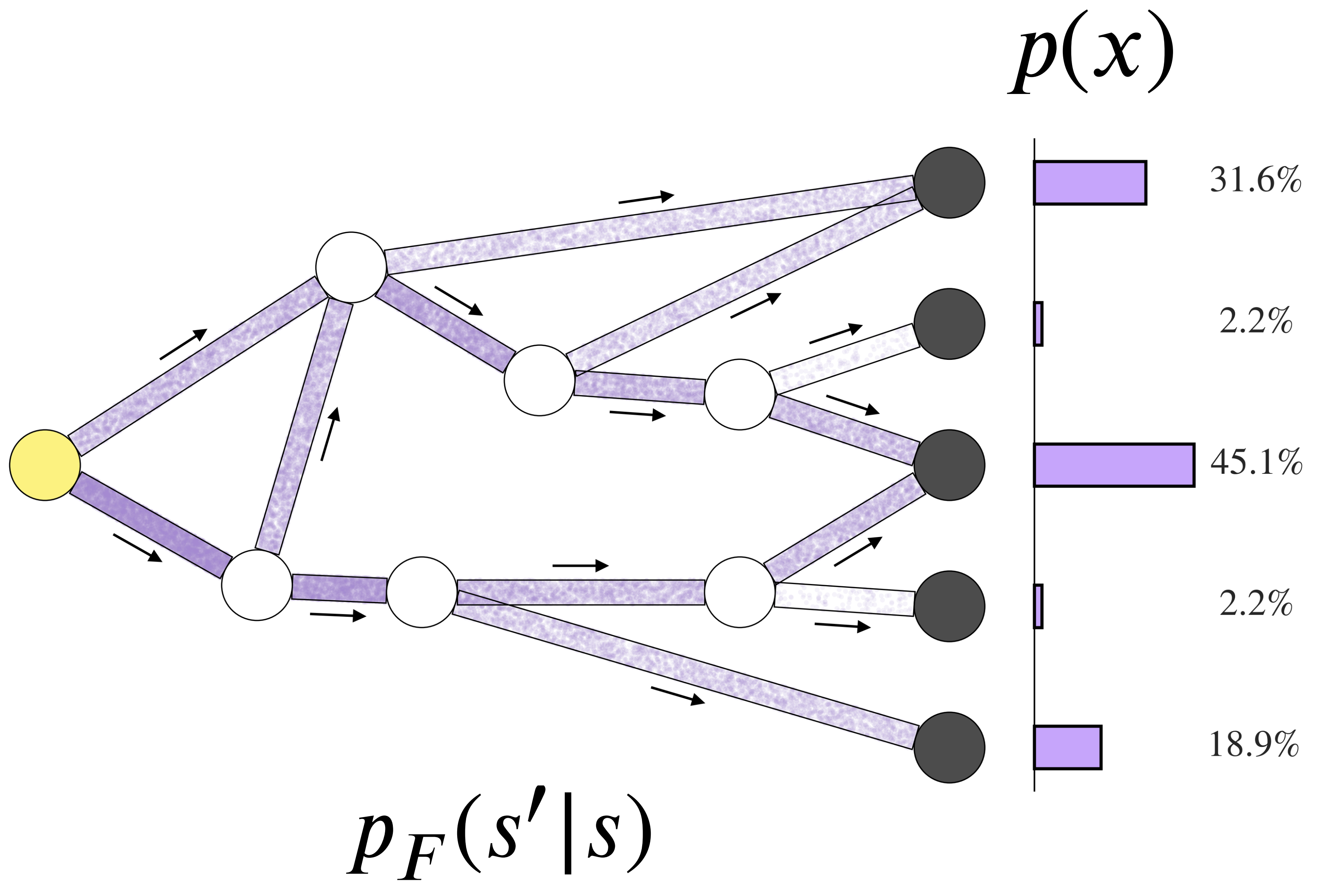
Proposition (GFlowNet mixture policy)**.**
Suppose distributions $p_1(x), \ldots, p_m(x)$ are realized by GFlowNets with forward policies $p_{1,F}(\cdot|\cdot), \ldots, p_{m,F}(\cdot|\cdot)$. Then, the mixture distribution $p_M(x) = \sum_{i=1}^m \omega_i p_i(x)$ with $\omega_1, \ldots, \omega_m \geq 0$ and $\sum_{i=1}^m \omega_i = 1$ is realized by the GFlowNet forward policy

$$p_{M,F}(s'|s) = \sum_{i=1}^{m} p(y = i|s) p_{i,F}(s'|s),$$

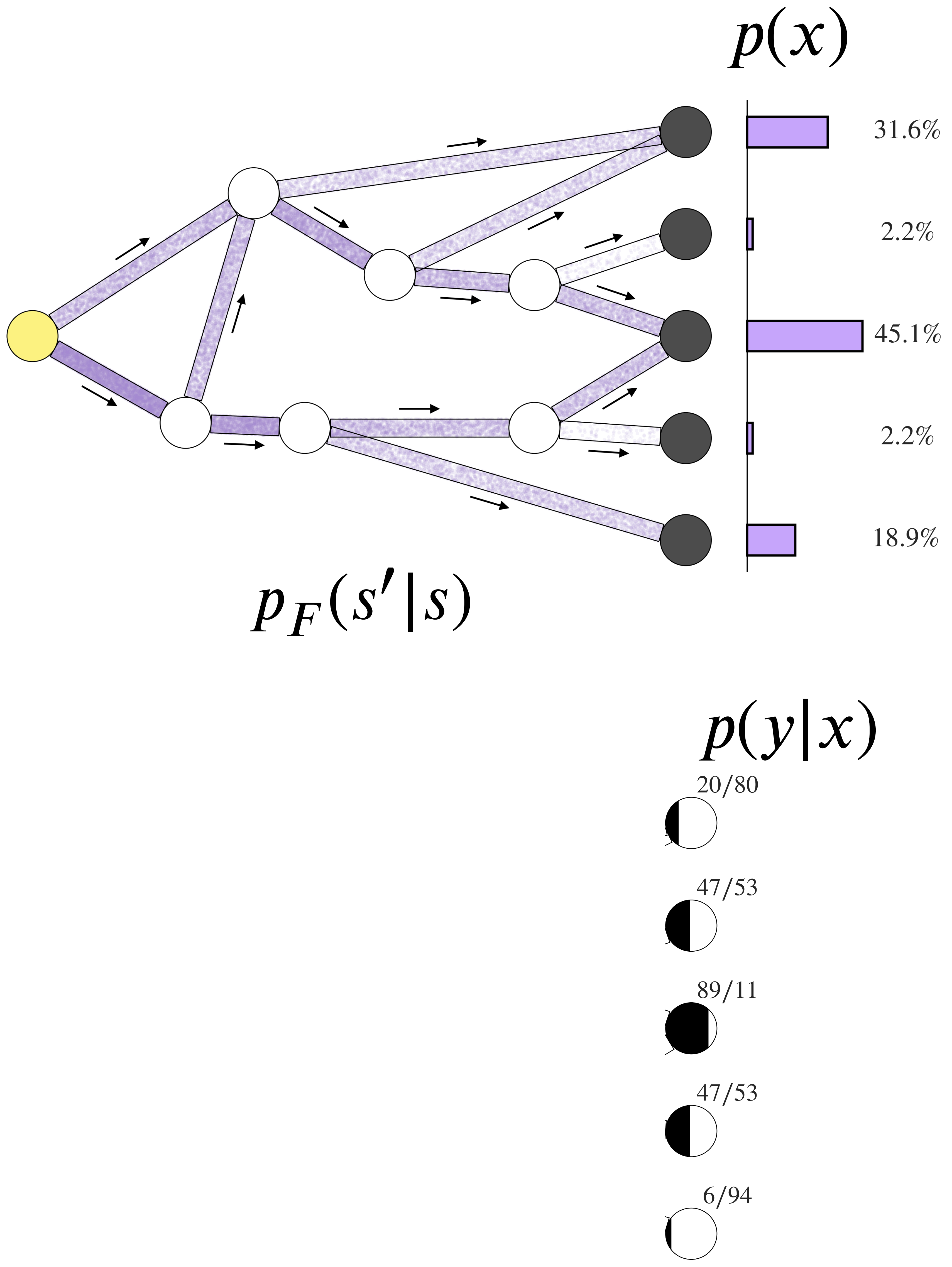
where y is a random variable such that the joint distribution of a GFlowNet trajectory τ and y is given by $p(\tau, y{=}i) = \omega_i p_i(\tau)$ for $i \in \{1, \ldots, m\}$.

New result for GFlowNets!

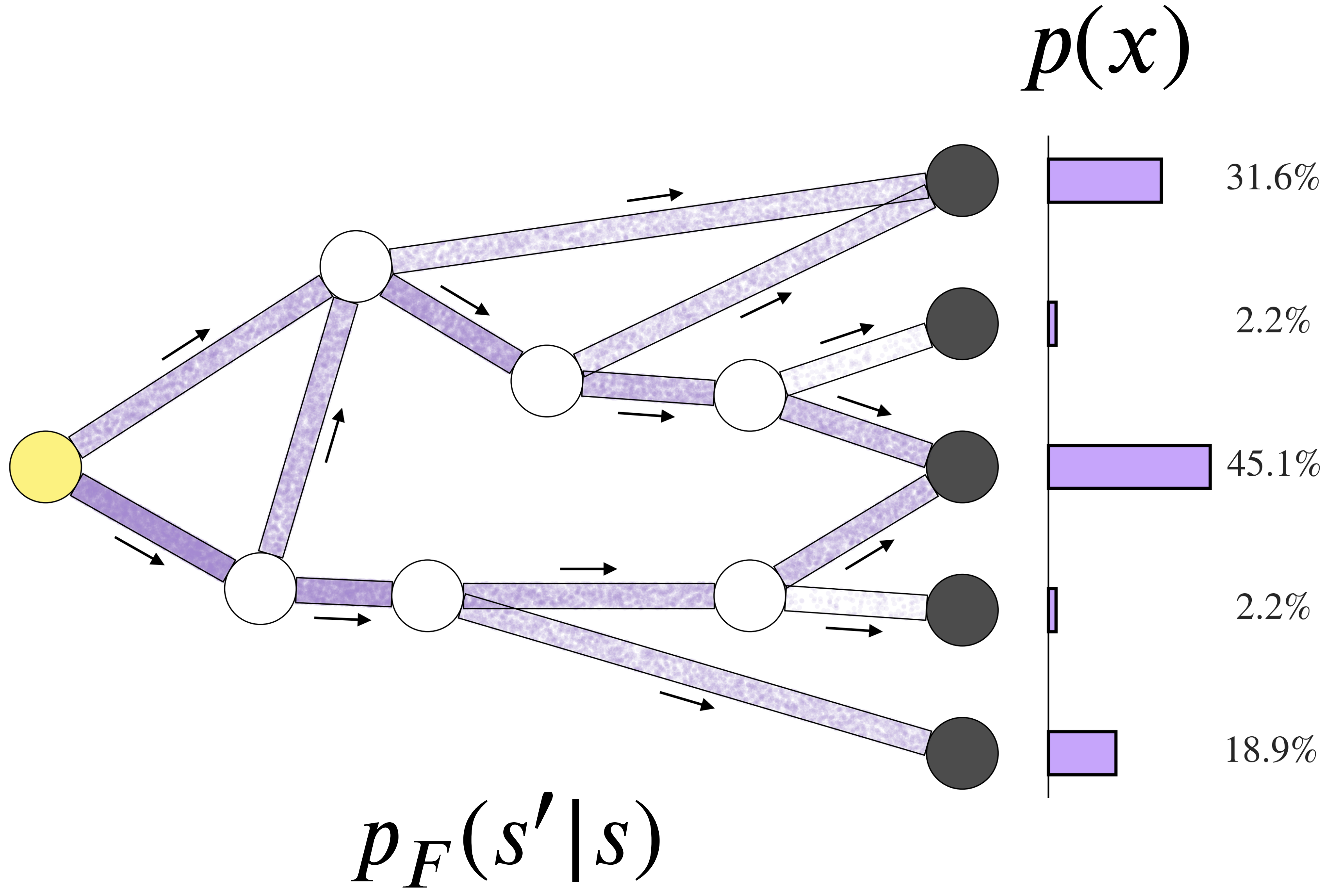
Composition Tools: Classifier Guidance

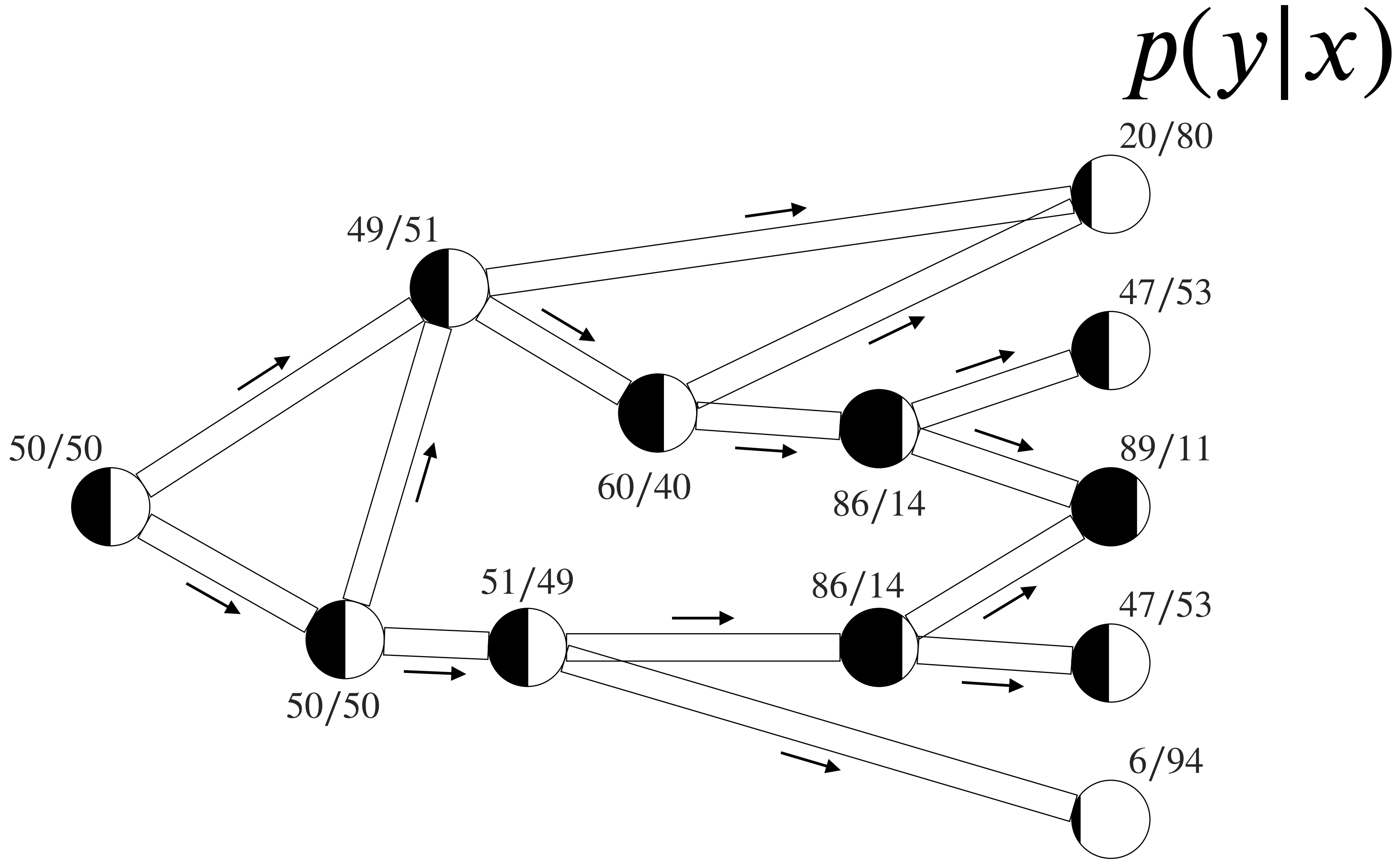


Composition Tools: Classifier Guidance



Composition Tools: Classifier Guidance

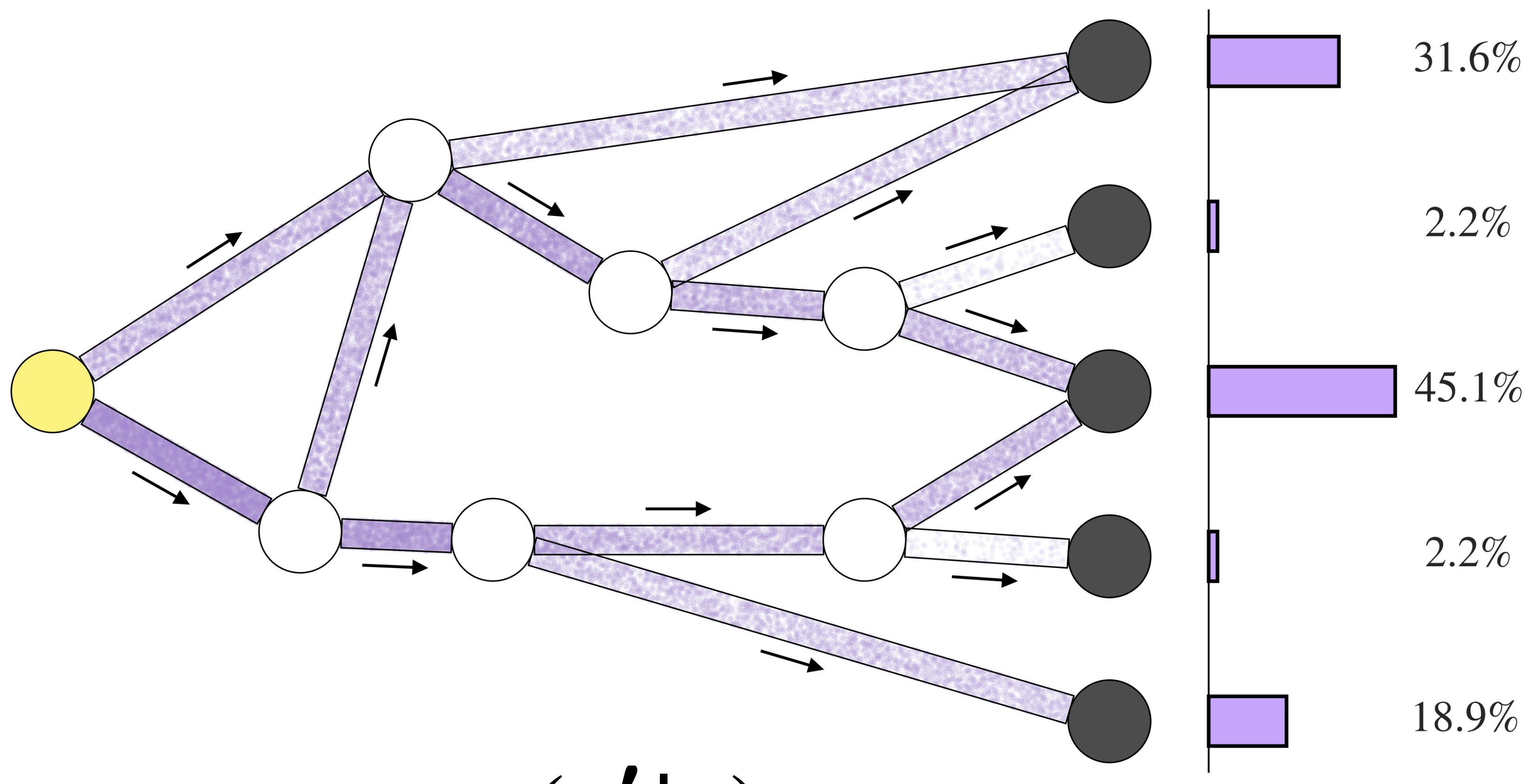




$$p(y|s)\ :\ p(\tau, x, y) = p(\tau, x)p(y|x)$$

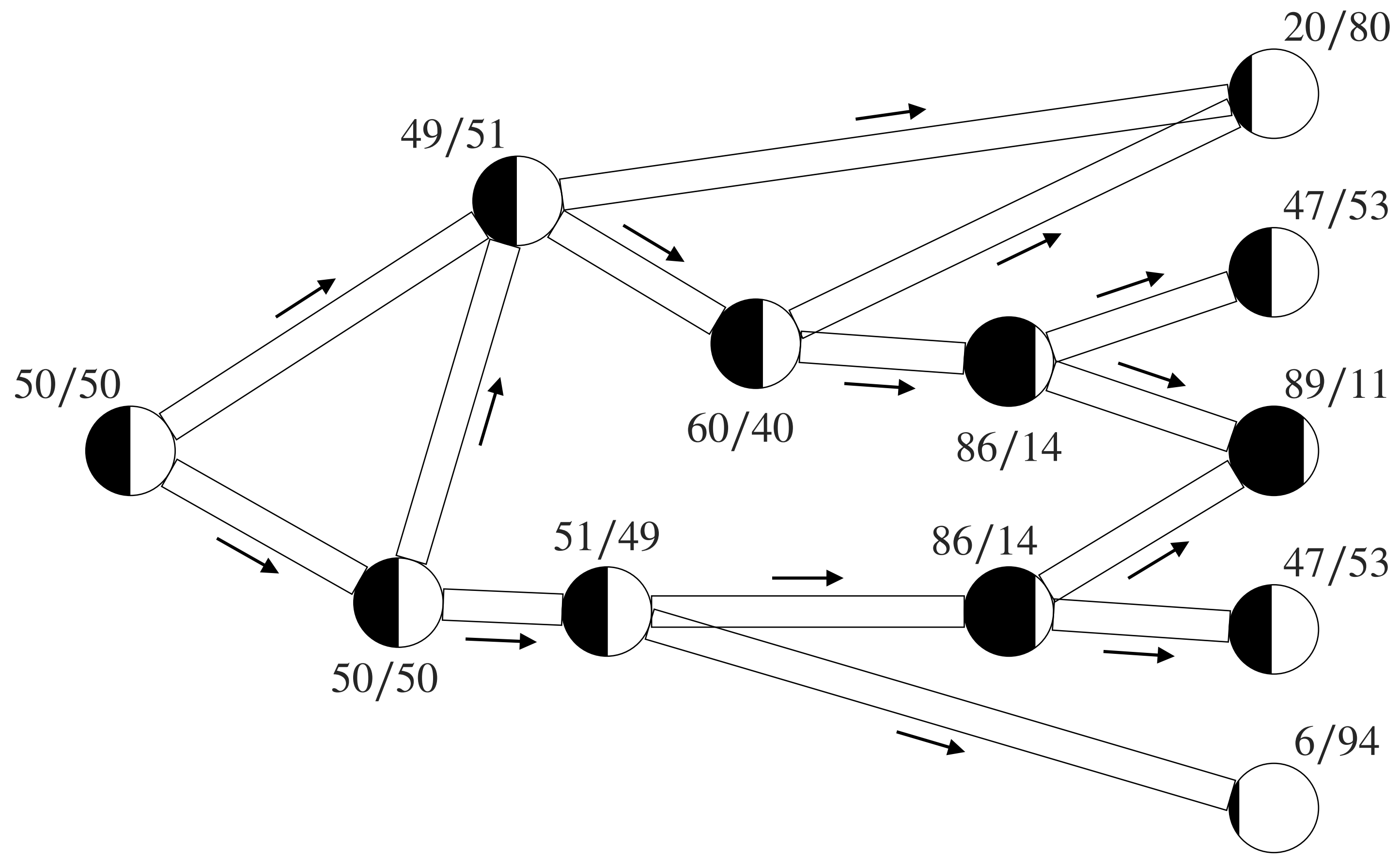
Composition Tools: Classifier Guidance

$p(x)$



$p_F(s'|s)$

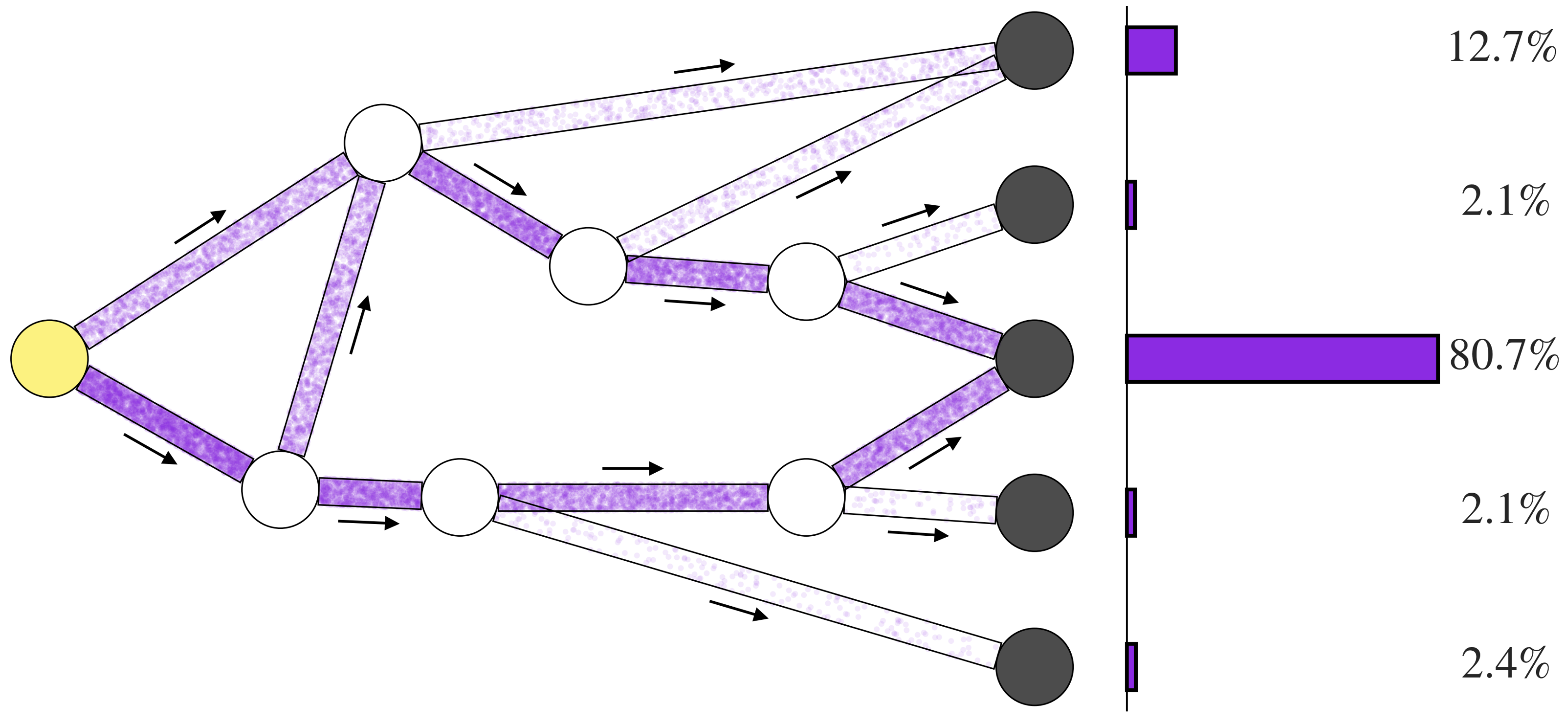
$p(y|x)$



$$p(y|s)\ :\ p(\tau, x, y) = p(\tau, x)p(y|x)$$

Classifier-guided GFlowNet

$p(x|y)$



$$p_F(s'|s, y) = p_F(s'|s)\frac{p(y|s')}{p(y|s)}$$

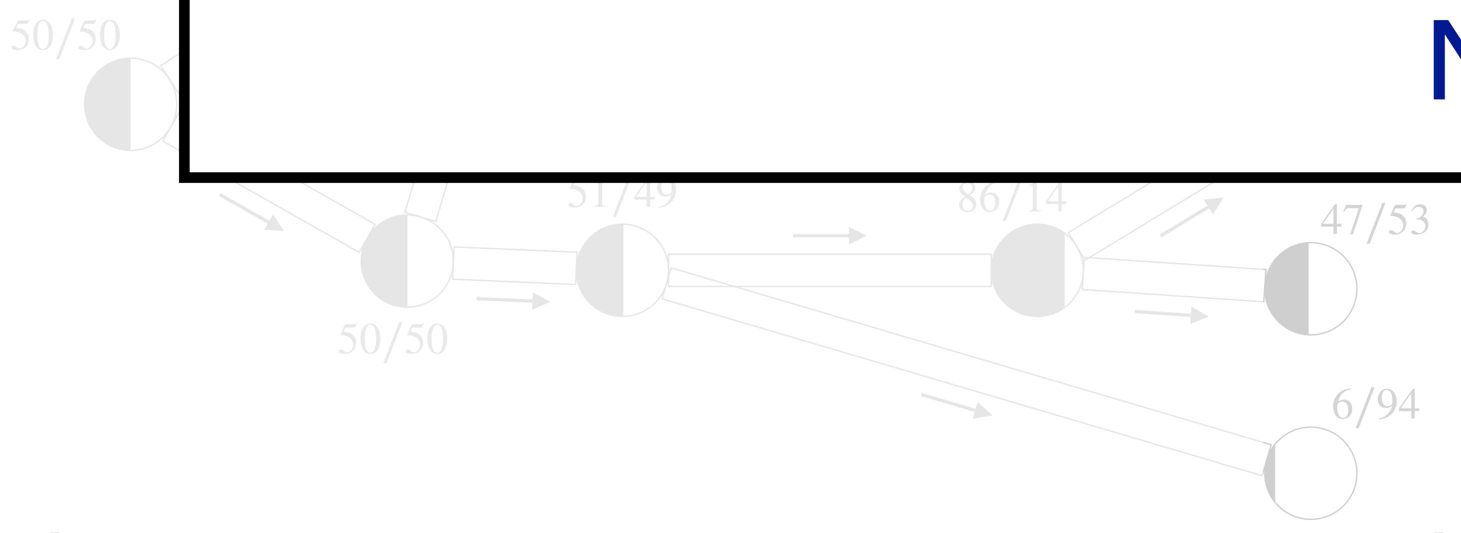
Composition Tools: Classifier Guidance

Proposition (GFlowNet classifier guidance)**.**
Consider a joint distribution $p(x, y)$ over a discrete space $\mathcal{X} \times \mathcal{Y}$ such that the marginal distribution $p(x)$ is realized by a GFlowNet with forward policy $p_F(\cdot|\cdot)$. Further, assume that the joint distribution of x, y, and GFlowNet trajectories $\tau = (s_0 \to \ldots \to s_n = x)$ decomposes as $p(\tau, x, y) = p(\tau, x)p(y|x)$, i.e. y is independent of the intermediate states $s_0, \ldots, s_{n-1}$ in τ given x. Then,

1. *For all non-terminal nodes $s \in \mathcal{S} \setminus \mathcal{X}$ in the GFlowNet DAG $(\mathcal{S}, \mathcal{A})$, the probabilities $p(y|s)$ satisfy*
$$p(y|s) = \sum_{s' : (s \to s') \in \mathcal{A}} p_F(s'|s)p(y|s').$$

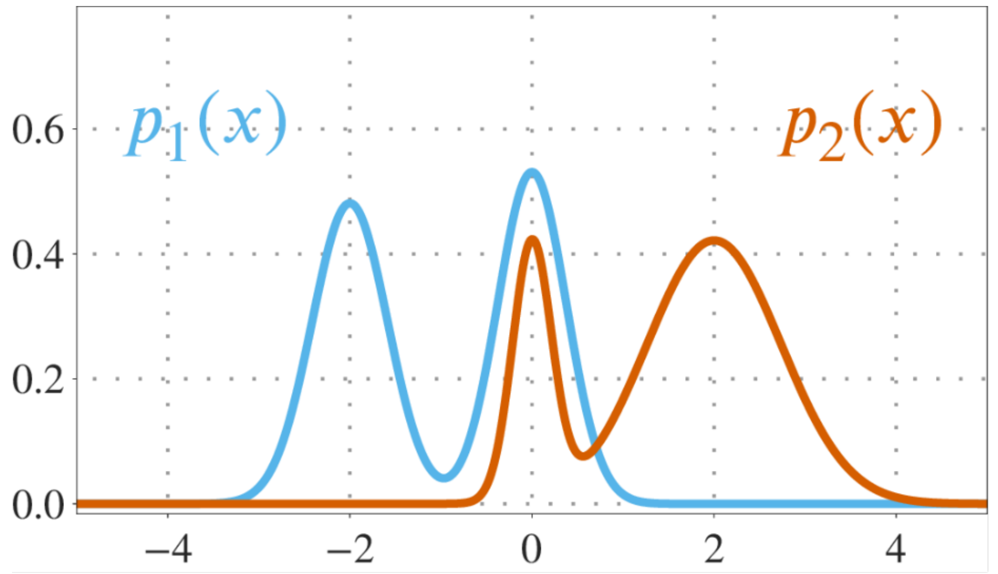
2. *The conditional distribution $p(x|y)$ is realized by the classifier-guided policy*
$$p_F(s'|s, y) = p_F(s'|s)\frac{p(y|s')}{p(y|s)}.$$

New result for GFlowNets!



$p(y|s) \; : \; p(\tau, x, y) = p(\tau, x)p(y|x)$

Composition Operations



Given: 2 distributions $p_1(x)$, $p_2(x)$

Prior

$$\widetilde{p}(x) = \frac{1}{2}p_1(x) + \frac{1}{2}p_2(x)$$

Observations

$$\widetilde{p}(y_k = i \mid x) = \frac{p_i(x)}{p_1(x) + p_2(x)}, \quad i \in \{1, 2\}$$

Posterior (composition)

$$\widetilde{p}(x \mid y_1 = i, y_2 = j) = \frac{p_i(x) p_j(x)}{p_1(x) + p_2(x)}$$

Composition Operations

Given: 2 distributions $p_1(x)$, $p_2(x)$

Prior

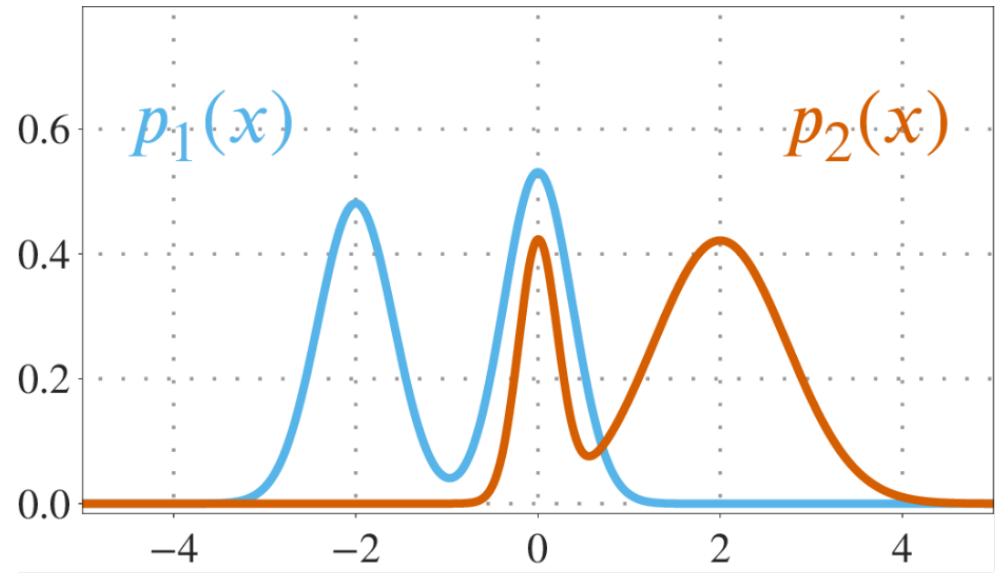
$$\widetilde{p}(x) = \frac{1}{2}p_1(x) + \frac{1}{2}p_2(x)$$

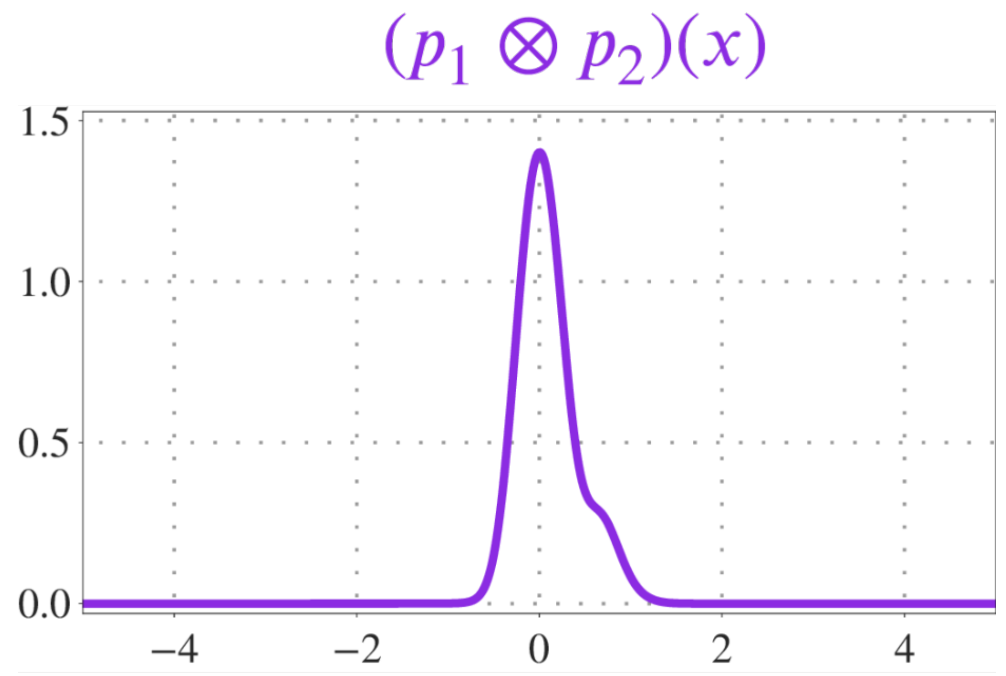
Observations

$$\widetilde{p}(y_k = i \mid x) = \frac{p_i(x)}{p_1(x) + p_2(x)}, \quad i \in \{1, 2\}$$

Posterior (composition)

$$\widetilde{p}(x \mid y_1 = i, y_2 = j) = \frac{p_i(x)p_j(x)}{p_1(x) + p_2(x)}$$





"Harmonic Mean"

$$(p_1 \otimes p_2)(x) = \widetilde{p}(x \mid y_1 = 1, y_2 = 2) \propto \frac{p_1(x)p_2(x)}{p_1(x) + p_2(x)}$$

Composition Operations

Given: 2 distributions $p_1(x)$, $p_2(x)$

Prior

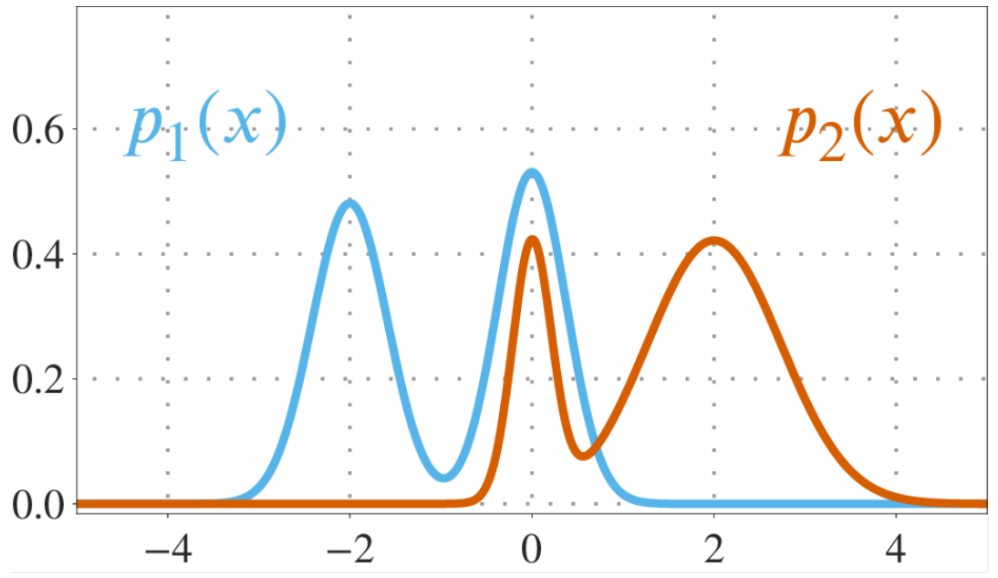
$$\widetilde{p}(x) = \frac{1}{2}p_1(x) + \frac{1}{2}p_2(x)$$

Observations

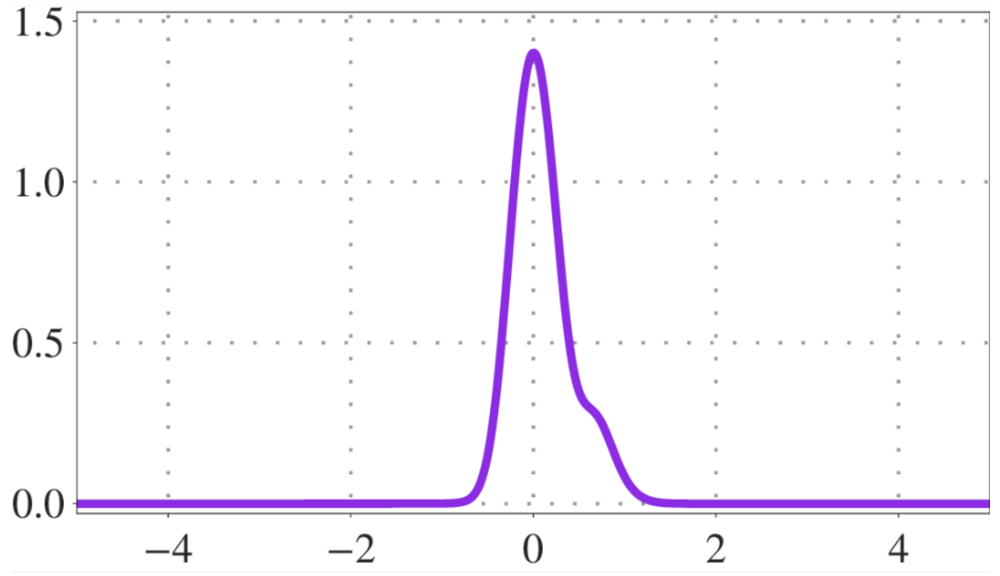
$$\widetilde{p}(y_k = i \mid x) = \frac{p_i(x)}{p_1(x) + p_2(x)}, \quad i \in \{1, 2\}$$

Posterior (composition)

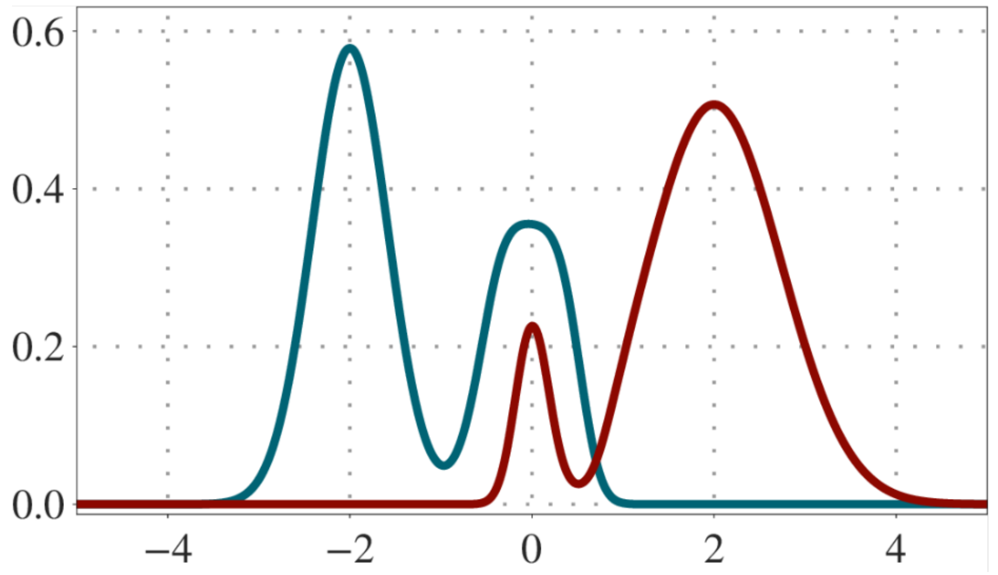
$$\widetilde{p}(x \mid y_1 = i, y_2 = j) = \frac{p_i(x)p_j(x)}{p_1(x) + p_2(x)}$$



$(p_1 \otimes p_2)(x)$



$(p_1 \RIGHTcircle p_2)(x)$ $(p_2 \RIGHTcircle p_1)(x)$



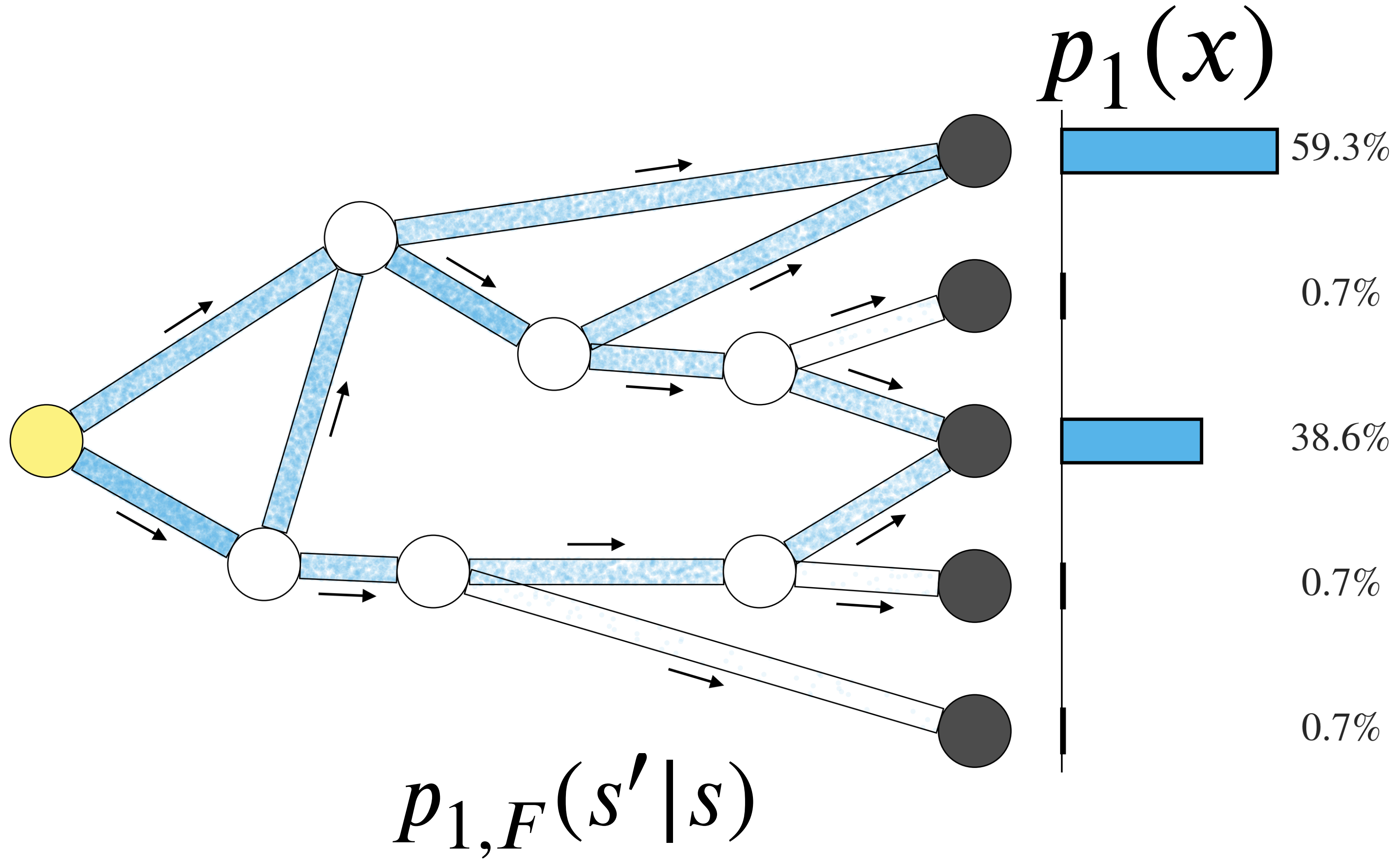
"Harmonic Mean"

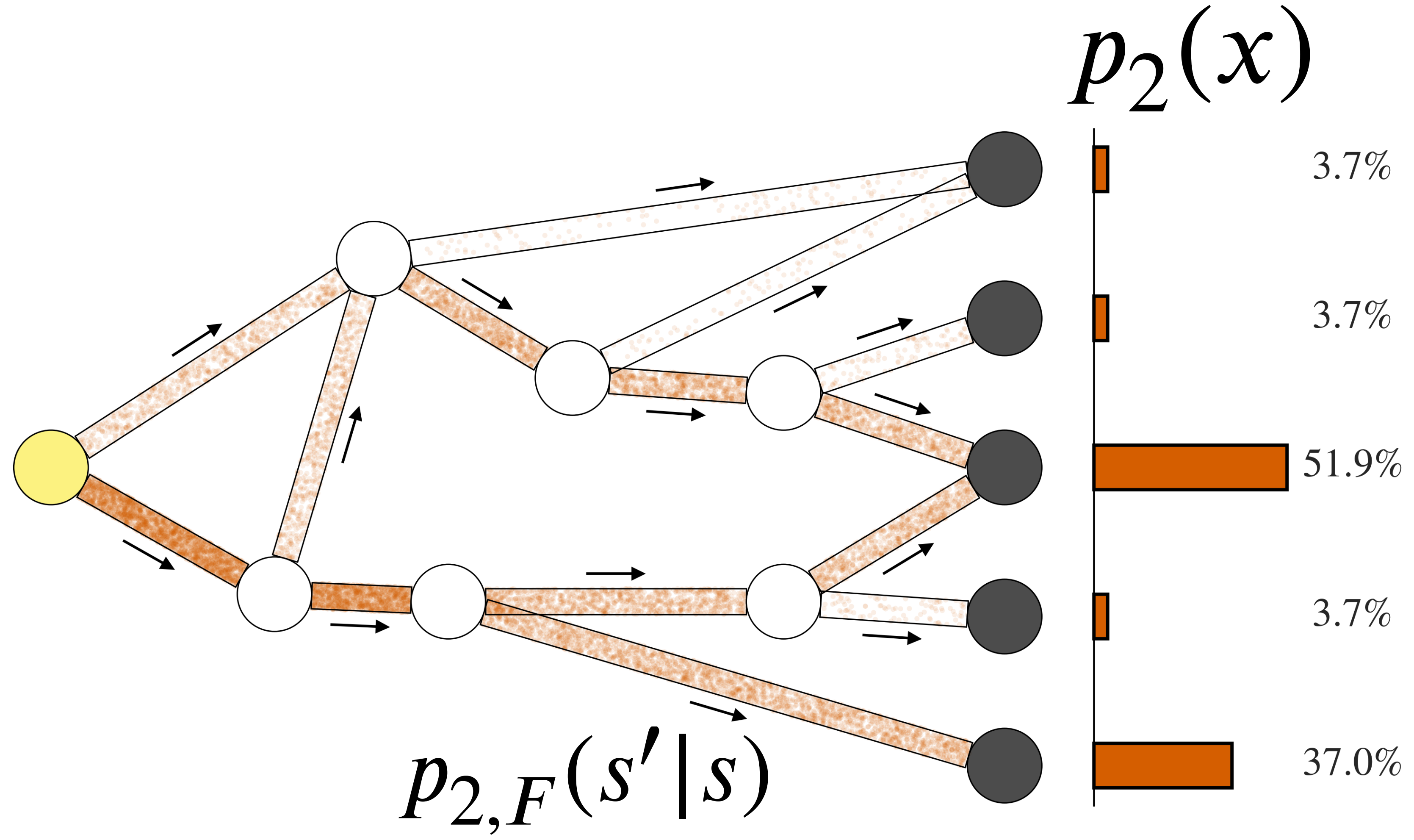
$$(p_1 \otimes p_2)(x) = \widetilde{p}(x \mid y_1 = 1, y_2 = 2) \propto \frac{p_1(x)p_2(x)}{p_1(x) + p_2(x)}$$

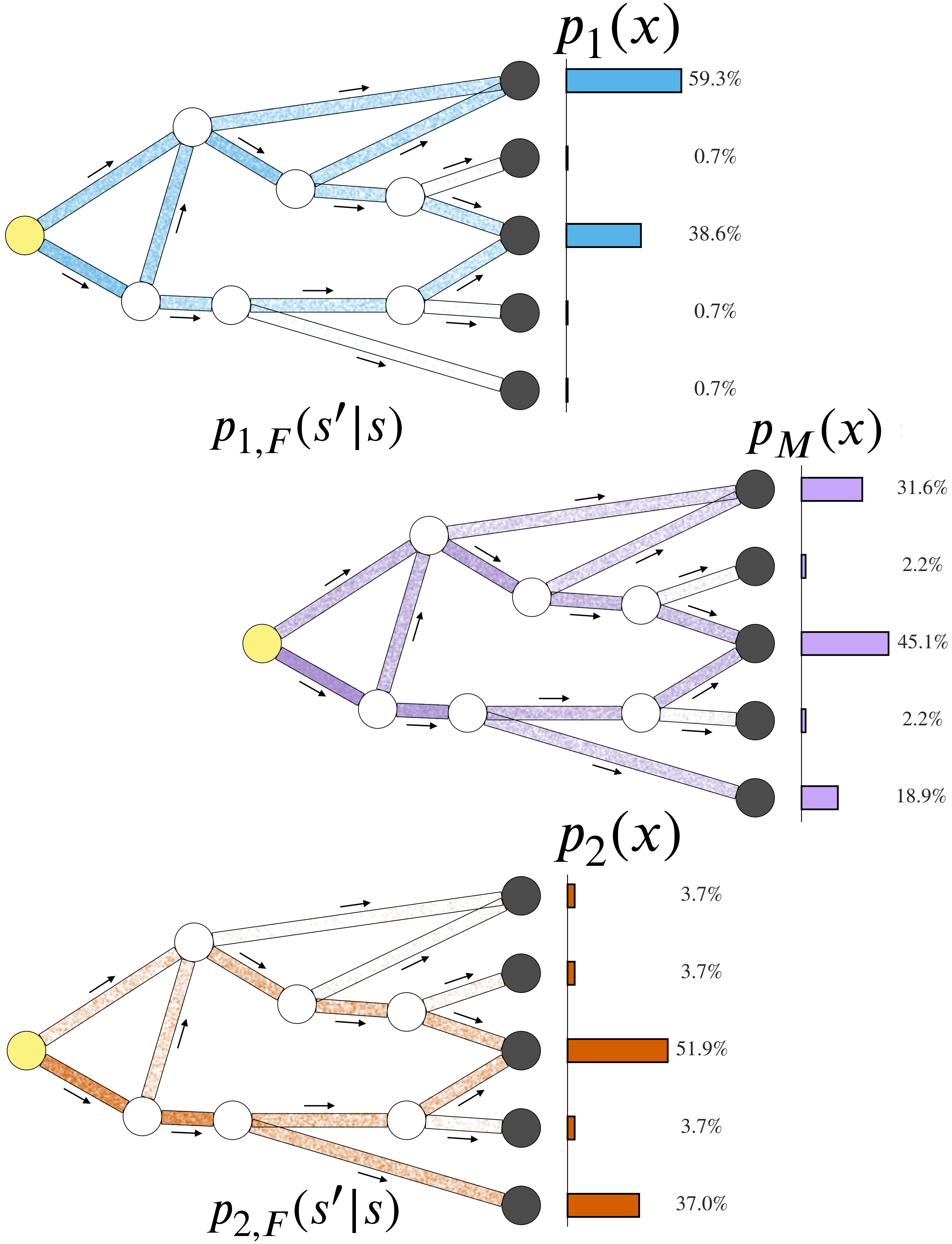
"Contrast"

$$(p_1 \RIGHTcircle p_2)(x) = \widetilde{p}(x \mid y_1 = 1, y_2 = 1) \propto \frac{\left(p_1(x)\right)^2}{p_1(x) + p_2(x)}$$

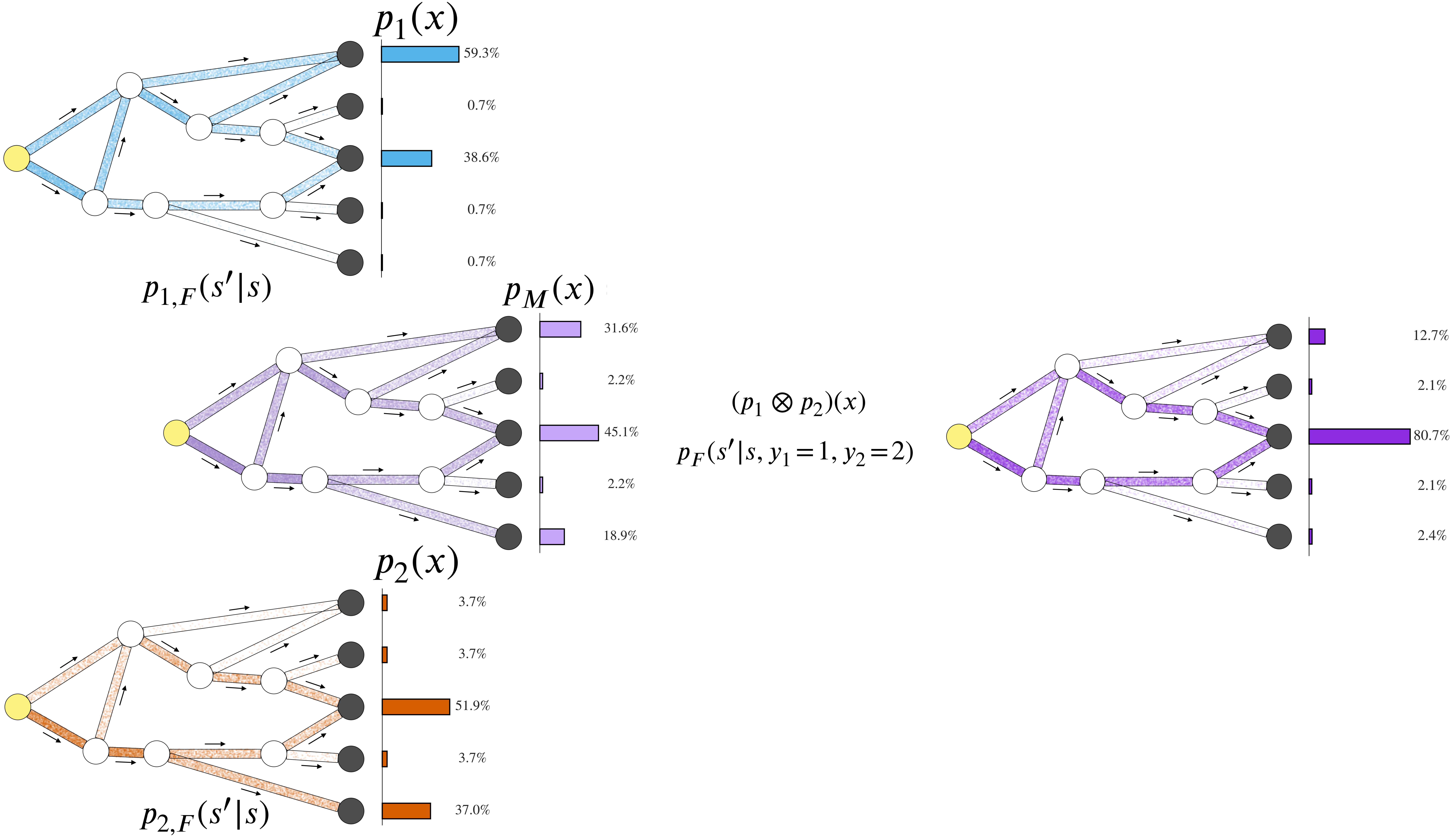
$$(p_2 \RIGHTcircle p_1)(x) = \widetilde{p}(x \mid y_1 = 2, y_2 = 2) \propto \frac{\left(p_2(x)\right)^2}{p_1(x) + p_2(x)}$$


$p_1(x)$
59.3%
0.7%
38.6%
0.7%
0.7%
$p_{1,F}(s'|s)$

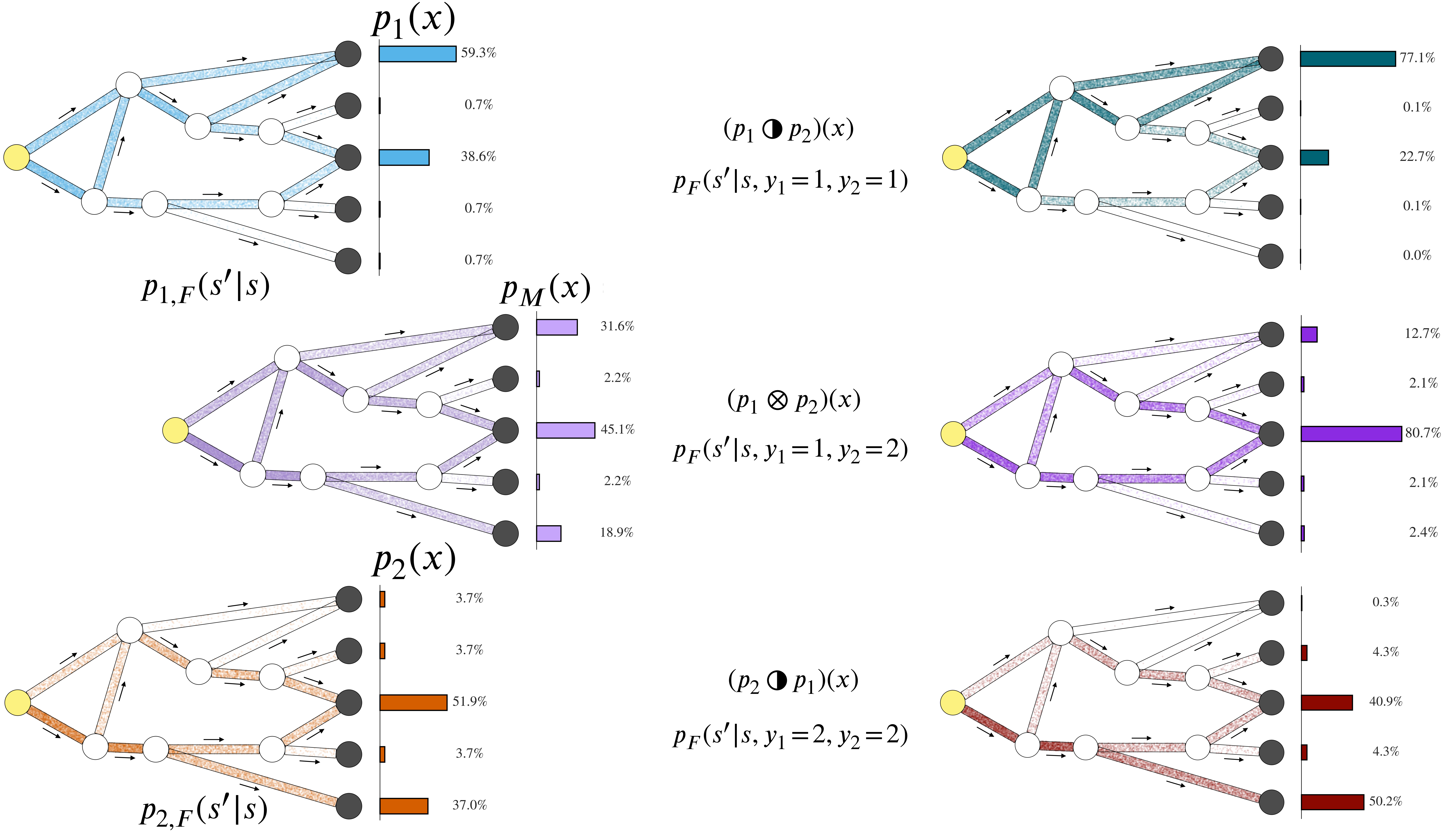

$p_2(x)$
3.7%
3.7%
51.9%
3.7%
37.0%
$p_{2,F}(s'|s)$



$p_1(x)$
59.3%
0.7%
38.6%
0.7%
0.7%
$p_{1,F}(s'|s)$
$p_M(x)$
31.6%
2.2%
45.1%
2.2%
18.9%
$p_2(x)$
3.7%
3.7%
51.9%
3.7%
37.0%
$p_{2,F}(s'|s)$

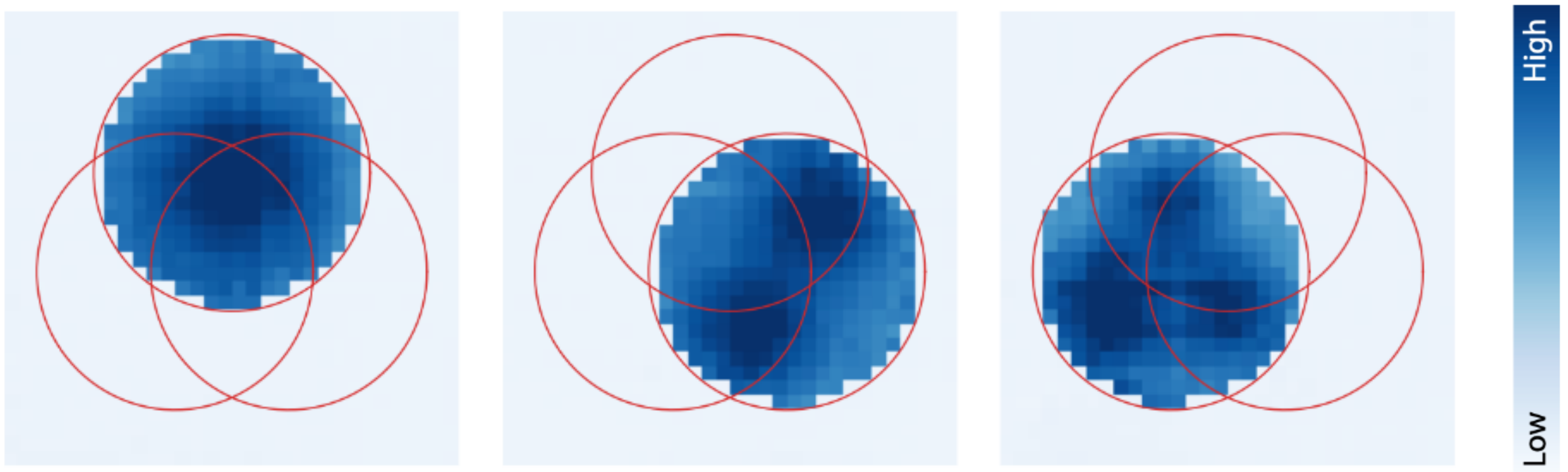


p1(x)
59.3%
0.7%
38.6%
0.7%
0.7%
p1,F(s′|s)
pM(x)
31.6%
2.2%
45.1%
2.2%
18.9%
p2(x)
3.7%
3.7%
51.9%
3.7%
37.0%
p2,F(s′|s)
(p1 ⊗ p2)(x)
pF(s′|s, y1 = 1, y2 = 2)
12.7%
2.1%
80.7%
2.1%
2.4%



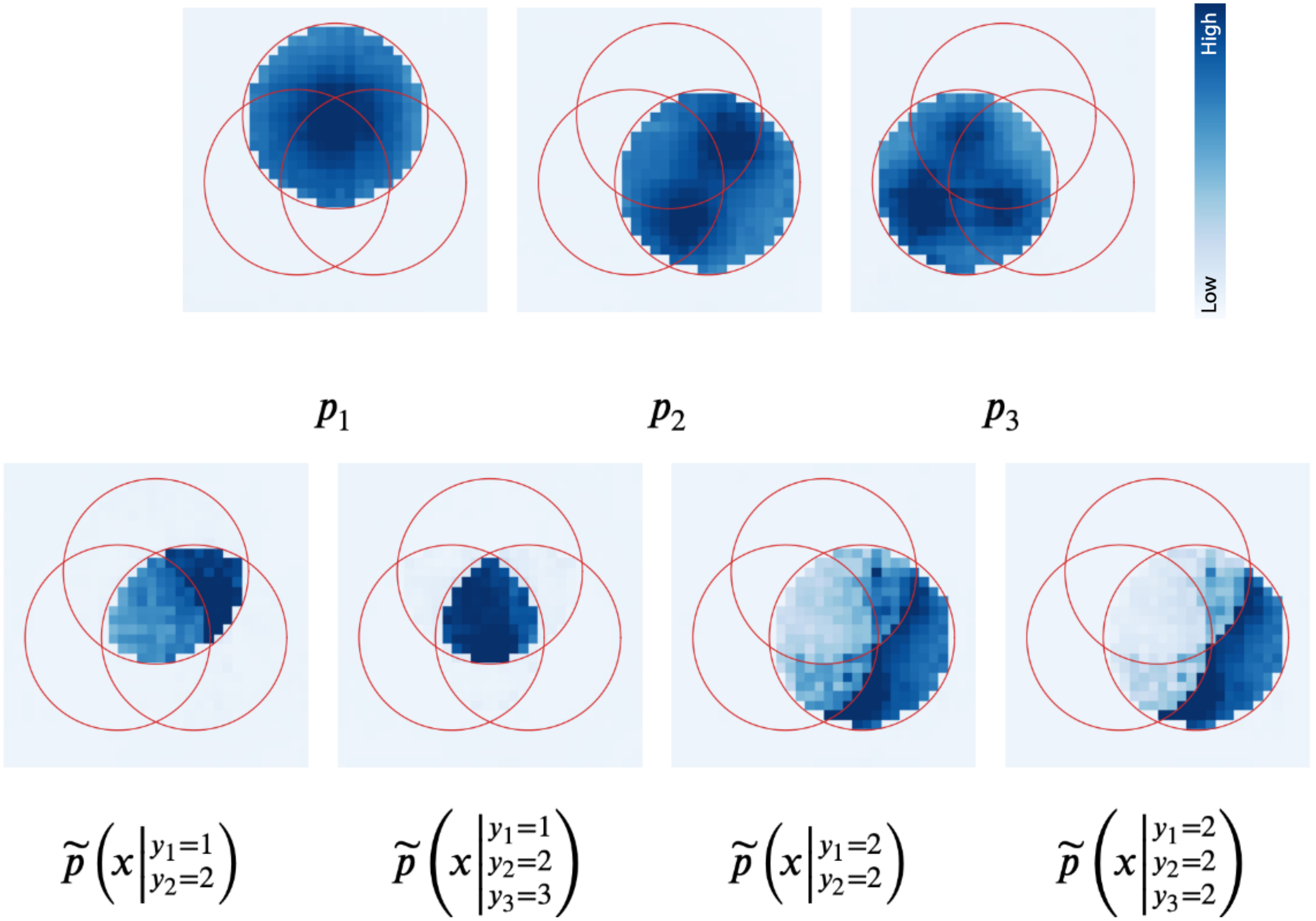
$p_1(x)$
59.3%
0.7%
38.6%
0.7%
0.7%
$p_{1,F}(s'|s)$
$(p_1 \circlerighthalfblack p_2)(x)$
$p_F(s'|s, y_1=1, y_2=1)$
77.1%
0.1%
22.7%
0.1%
0.0%
$p_M(x)$
31.6%
2.2%
45.1%
2.2%
18.9%
$(p_1 \otimes p_2)(x)$
$p_F(s'|s, y_1=1, y_2=2)$
12.7%
2.1%
80.7%
2.1%
2.4%
$p_2(x)$
3.7%
3.7%
51.9%
3.7%
37.0%
$p_{2,F}(s'|s)$
$(p_2 \circlerighthalfblack p_1)(x)$
$p_F(s'|s, y_1=2, y_2=2)$
0.3%
4.3%
40.9%
4.3%
50.2%

Composition Operations



p_1 p_2 p_3

Composition Operations



Composition Operations

Given: m distributions $p_1(x), \ldots, p_m(x)$

Prior

$$\widetilde{p}(x) = \frac{1}{m}\sum_{j=1}^{m} p_j(x)$$

Observations

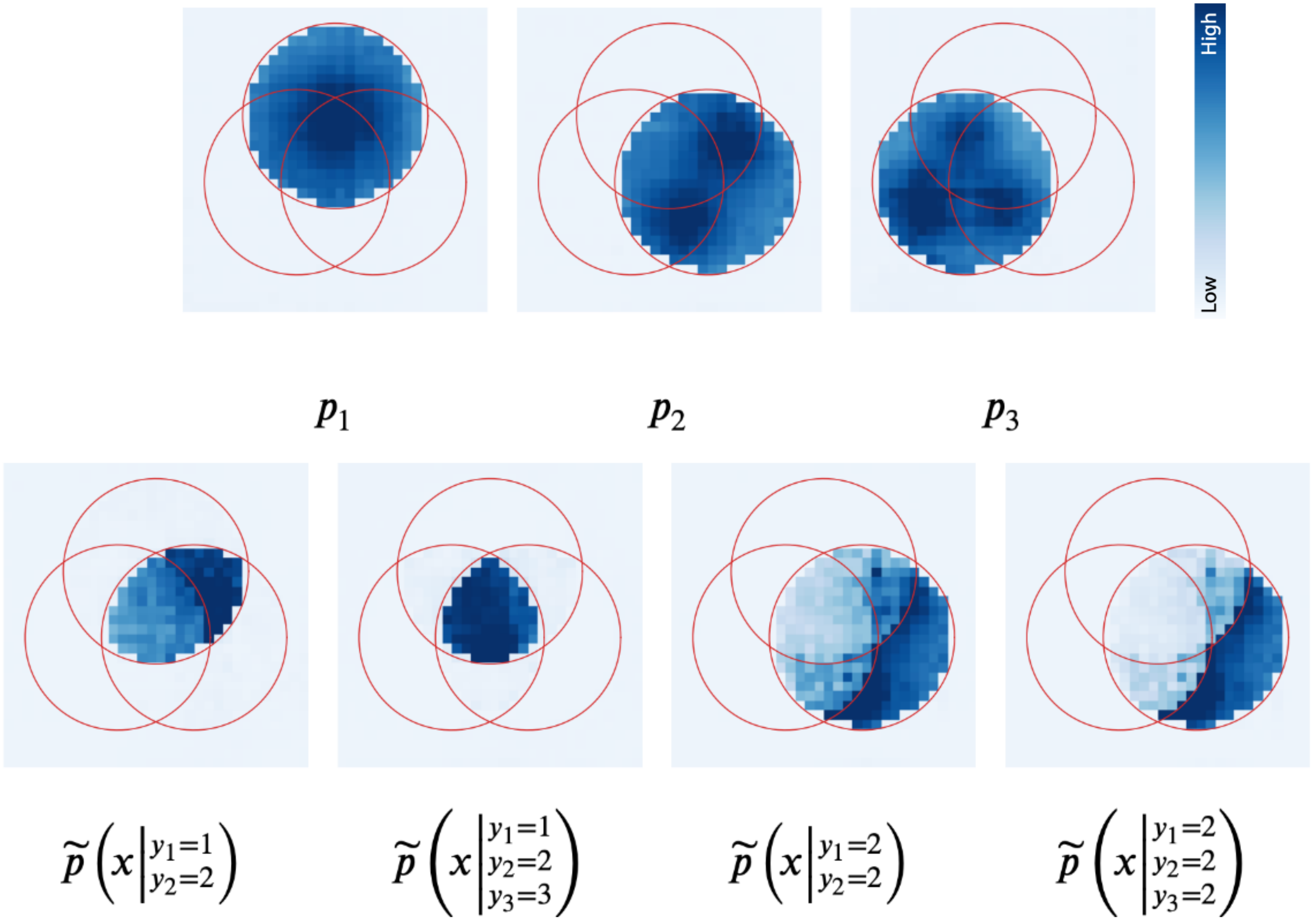
$$\widetilde{p}(y_k = i \mid x) = \frac{p_i(x)}{\sum_{j=1}^{m} p_j(x)}, \quad i \in \{1, \ldots, m\}$$

Joint

$$\widetilde{p}(x, y_1, \ldots, y_n) = \widetilde{p}(x)\prod_{k=1}^{n}\widetilde{p}(y_k \mid x)$$

Posterior (composition)

$$\widetilde{p}(x \mid y_1 = i_1, \ldots, y_n = i_n) \propto \frac{\prod_{k=1}^{n} p_{i_k}(x)}{\left(\sum_{j=1}^{m} p_j(x)\right)^{n-1}}$$



Composition Operations

Given: m distributions $p_1(x), \ldots, p_m(x)$

Prior

$$\widetilde{p}(x) = \frac{1}{m} \sum_{j=1}^{m} p_j(x)$$

Observations

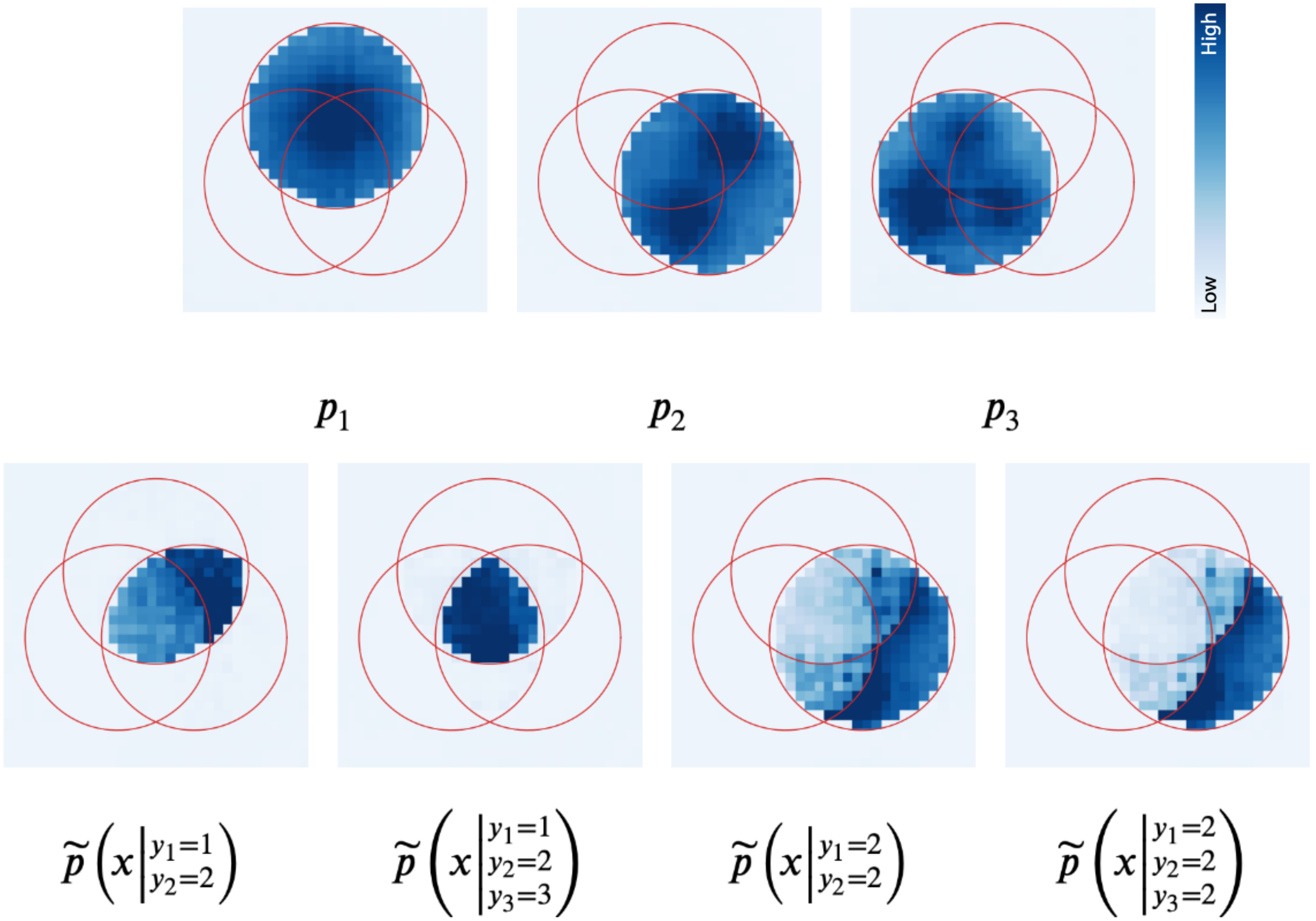
$$\widetilde{p}(y_k = i \mid x) = \frac{p_i(x)}{\sum_{j=1}^{m} p_j(x)}, \quad i \in \{1, \ldots, m\}$$

Joint

$$\widetilde{p}(x, y_1, \ldots, y_n) = \widetilde{p}(x) \prod_{k=1}^{n} \widetilde{p}(y_k \mid x)$$

Posterior (composition)

$$\widetilde{p}(x \mid y_1 = i_1, \ldots, y_n = i_n) \propto \frac{\prod_{k=1}^{n} p_{i_k}(x)}{\left(\sum_{j=1}^{m} p_j(x)\right)^{n-1}}$$



Control of resulting distribution via observations

+ more tools for control in the thesis

Theorem.
Suppose distributions $p_1(x), \ldots, p_m(x)$ are realized by GFlowNets with forward policies $p_{1,F}(\cdot|\cdot), \ldots, p_{m,F}(\cdot|\cdot)$ respectively. Let $y_1, \ldots, y_n$ be random variables defined by

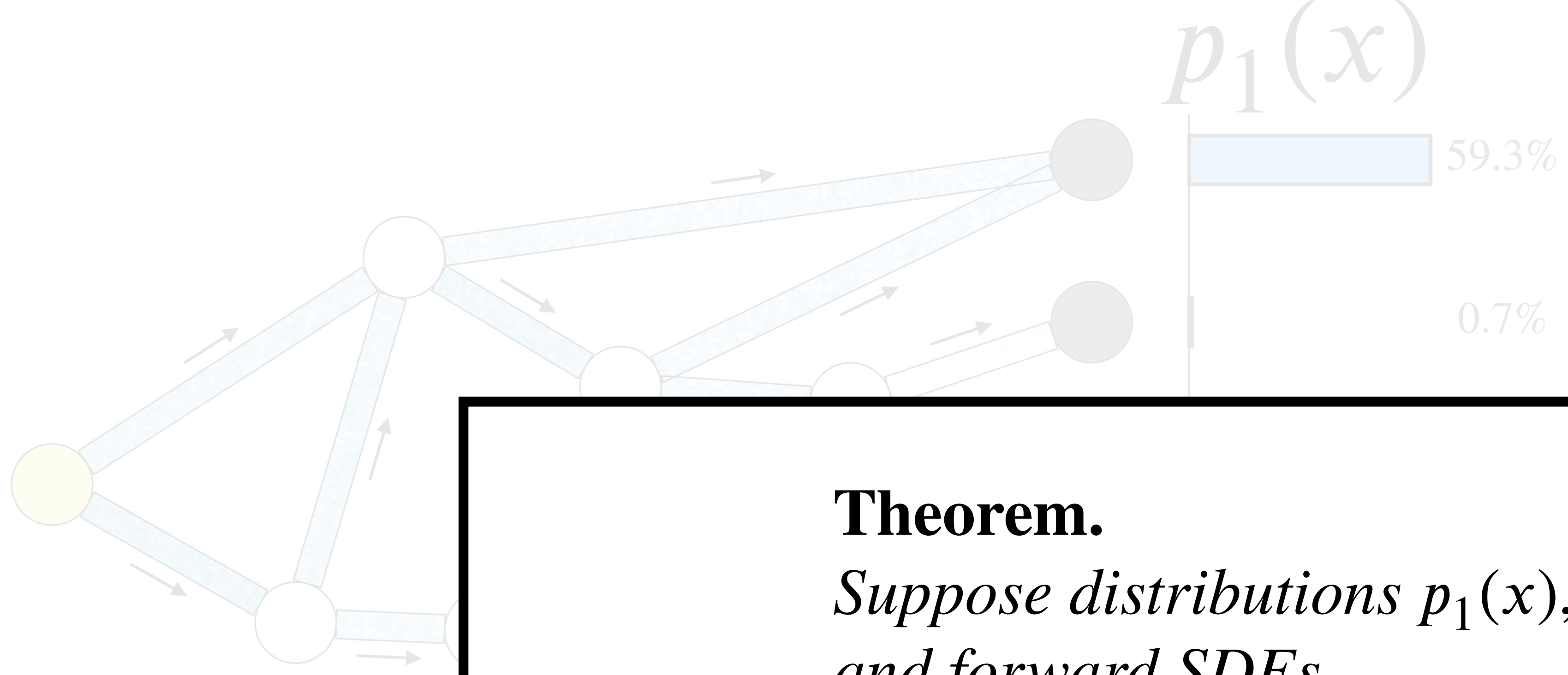
$$\widetilde{p}(x, y_1, \ldots, y_n) = \widetilde{p}(x) \prod_{k=1}^{n} \widetilde{p}(y_k|x), \qquad \widetilde{p}(x) = \frac{1}{m} \sum_{i=1}^{m} p_i(x),$$

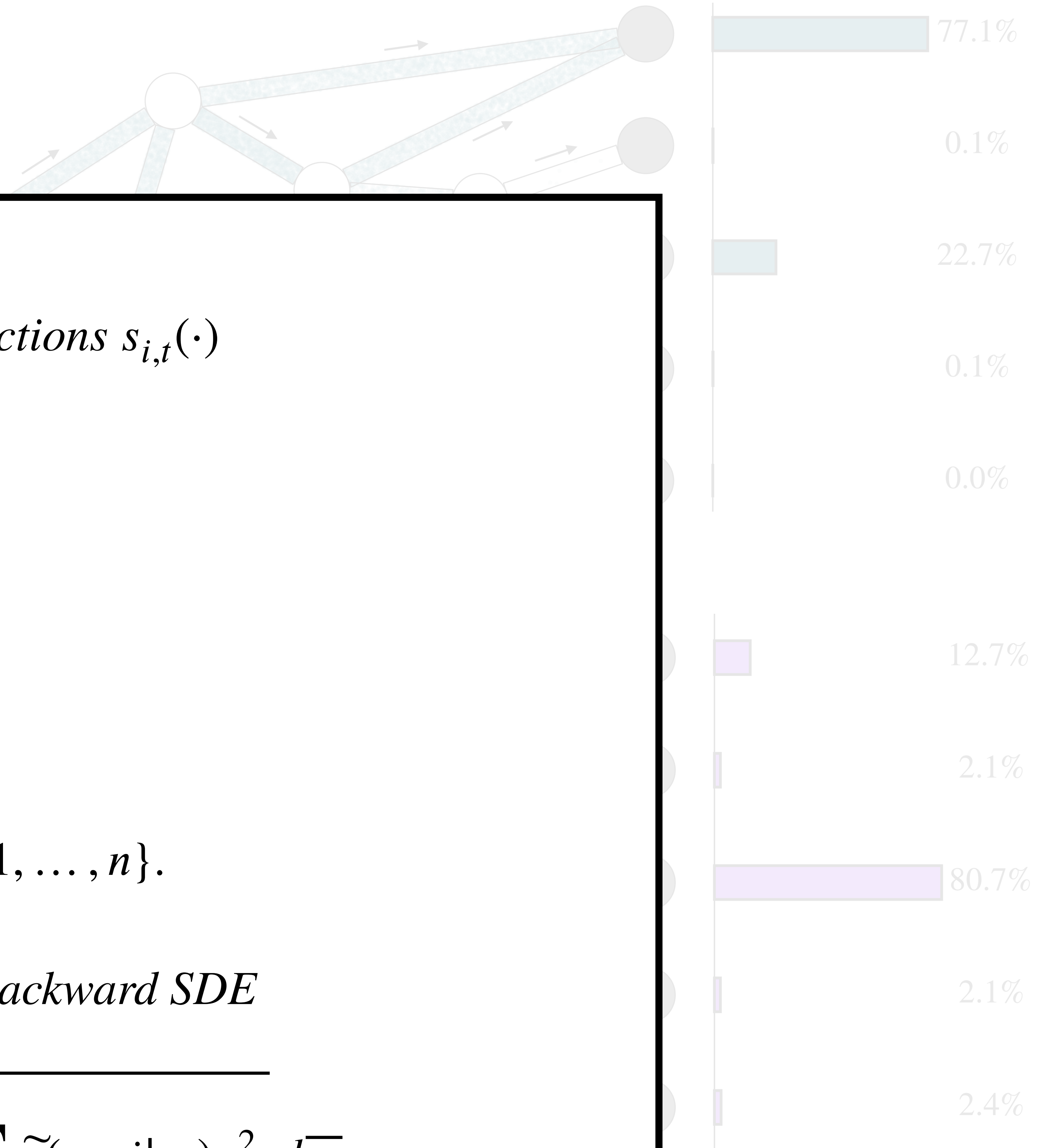
$$\widetilde{p}(y_k = i) = \frac{1}{m} \quad \forall k \in \{1, \ldots, n\}, \qquad \widetilde{p}(y_k = i|x) = \frac{p_i(x)}{\sum_{j=1}^{m} p_j(x)} \quad \forall k \in \{1, \ldots, n\}.$$

Then, the conditional $\widetilde{p}(x|y_1, \ldots, y_n)$ is realized by the forward policy

$$p_F(s'|s, y_1, \ldots, y_n) = \frac{\widetilde{p}(y_1, \ldots, y_n|s')}{\widetilde{p}(y_1, \ldots, y_n|s)} \sum_{i=1}^{m} p_{i,F}(s'|s) \widetilde{p}(y = i|s)$$

New result for GFlowNets!





Theorem.
Suppose distributions $p_1(x), \ldots, p_m(x)$ are realized by diffusion models with score functions $s_{i,t}(\cdot)$ and forward SDEs

$$dx_{i,t} = f_{i,t}(x_{i,t})\,dt + g_{i,t}\,dw_{i,t}, \qquad i \in \{1, \ldots, n\}.$$

Let $y_1, \ldots, y_n$ be random variables defined by

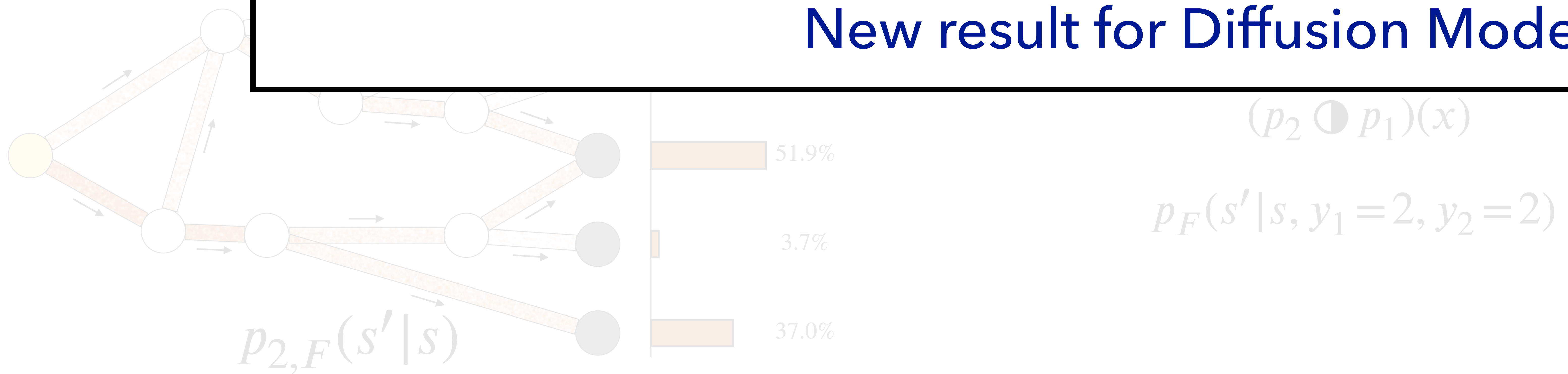
$$\widetilde{p}(x, y_1, \ldots, y_n) = \widetilde{p}(x) \prod_{k=1}^{n} \widetilde{p}(y_k | x), \qquad \widetilde{p}(x) = \frac{1}{m} \sum_{i=1}^{m} p_i(x),$$

$$\widetilde{p}(y_k = i) = \frac{1}{m} \quad \forall k \in \{1, \ldots, n\}, \qquad \widetilde{p}(y_k = i | x) = \frac{p_i(x)}{\sum_{j=1}^{m} p_j(x)} \quad \forall k \in \{1, \ldots, n\}.$$

Then, the conditional $\widetilde{p}(x | y_1, \ldots, y_n)$ is realized by a classifier-guided diffusion with backward SDE

$$dx_t = \left[\sum_{i=1}^{m} \widetilde{p}(y = i | x_t)\Big(f_{i,t}(x_t) - g_{i,t}^2\Big(s_{i,t}(x_t) + \nabla_{x_t} \log \widetilde{p}(y_1, \ldots, y_n | x_t)\Big)\Big)\right] dt + \sqrt{\sum_{i=1}^{m} \widetilde{p}(y = i | x_t) g_{i,t}^2}\, d\overline{w}_t.$$

New result for Diffusion Models!

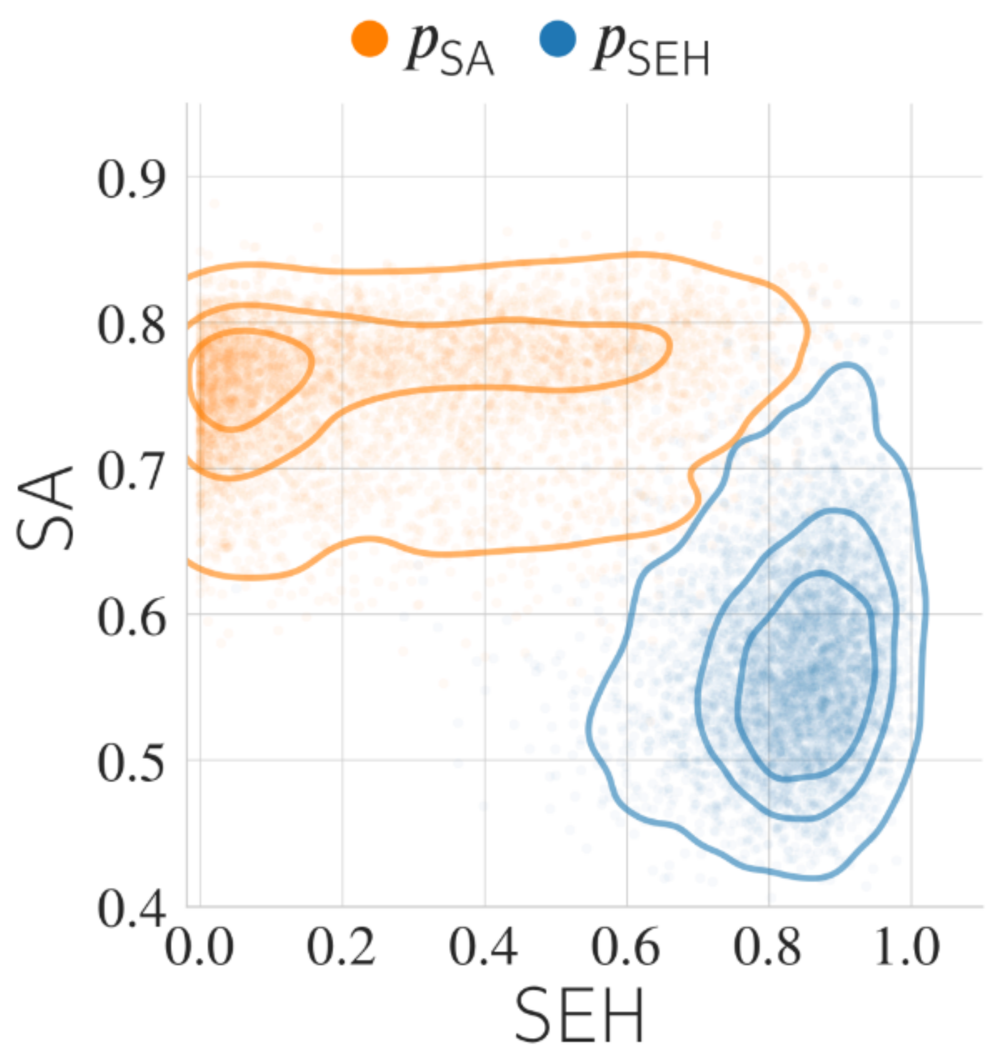


	Models	Composition Operations	Sampling Algorithm
[Hinton, Neural Computation 2002] [Du et al, NeurIPS 2020]	Energy-based models (EBMs) $p_i(x) \propto \exp(-E_i(x;\theta))$	Principle: energy-function arithmetic Product: $\frac{1}{Z}\, p_1(x)\, p_2(x)$ Negation: $\frac{1}{Z}\, \frac{p_1(x)}{\left(p_2(x)\right)^{\gamma}}$	MCMC Langevin dynamics
[Liu et al, ECCV 2022] [Du et al, ICML 2023]	Diffusion models $p_i(x):\ s_{i,t}(x_t;\theta) \approx \nabla_{x_t} \log p_{i,t}(x_t)$	Principle: score-function arithmetic Product: $\frac{1}{Z}\, p_1(x)\, p_2(x)$ Negation: $\frac{1}{Z}\, \frac{p_1(x)}{\left(p_2(x)\right)^{\gamma}}$	Diffusion sampling + annealed MCMC

Challenge: iterative generative processes (Diffusion models & GFlowNets) impose delicate balance conditions

	Models	Composition Operations	Sampling Algorithm
[Hinton, Neural Computation 2002] [Du et al, NeurIPS 2020]	Energy-based models (EBMs) $p_i(x) \propto \exp(-E_i(x;\theta))$	Principle: energy-function arithmetic Product: $\frac{1}{Z} p_1(x)\, p_2(x)$ Negation: $\frac{1}{Z} \frac{p_1(x)}{\left(p_2(x)\right)^\gamma}$	MCMC Langevin dynamics
[Liu et al, ECCV 2022] [Du et al, ICML 2023]	Diffusion models $p_i(x):\ s_{i,t}(x_t;\theta) \approx \nabla_{x_t} \log p_{i,t}(x_t)$	Principle: score-function arithmetic Product: $\frac{1}{Z} p_1(x)\, p_2(x)$ Negation: $\frac{1}{Z} \frac{p_1(x)}{\left(p_2(x)\right)^\gamma}$	Diffusion sampling + annealed MCMC
Challenge: iterative generative processes (Diffusion models & GFlowNets) impose delicate balance conditions			
Compositional Sculpting (ours)	Diffusion models $p_i(x):\ s_{i,t}(x_t;\theta) \approx \nabla_{x_t} \log p_{i,t}(x_t)$ GFlowNets $p_i(x):\ p_{i,F}(s_{t+1} \vert s_t;\theta)$	Principle: mixture & conditional generative processes **Harmonic Mean**: $\frac{1}{Z} \frac{p_1(x)\, p_2(x)}{p_1(x) + p_2(x)}$ **Contrast**: $\frac{1}{Z} \frac{\left(p_1(x)\right)^2}{p_1(x) + p_2(x)}$ $(+)$ **other operations**	Diffusion mixture + classifier guidance GFlowNet mixture + classifier guidance

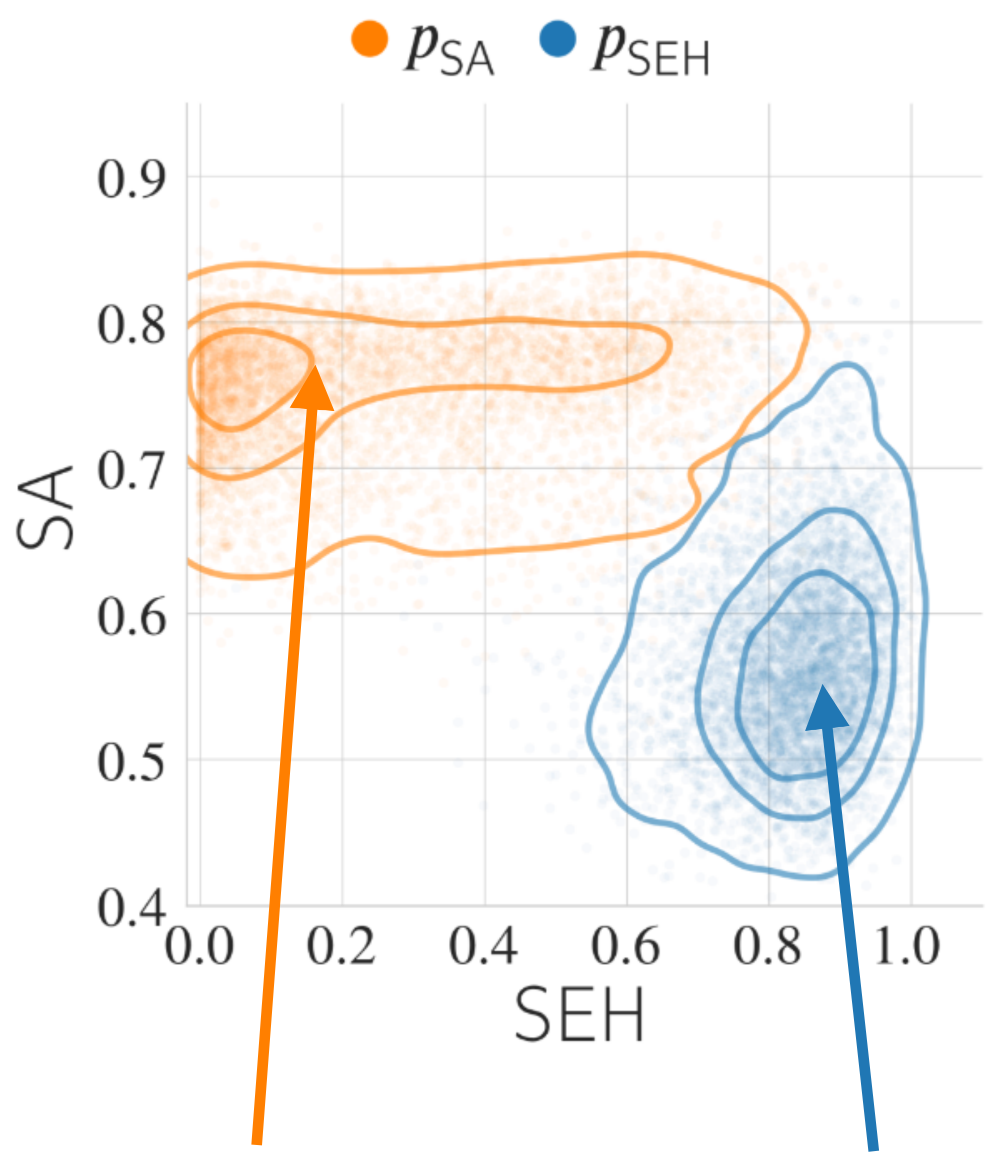
Results: GFlowNet Composition For Molecule Generation



Molecular property scores (“Rewards”)

- SEH – learned proxy of a protein binding score (soluble epoxide hydrolase)
- SA – synthetic availability score

Results: GFlowNet Composition For Molecule Generation



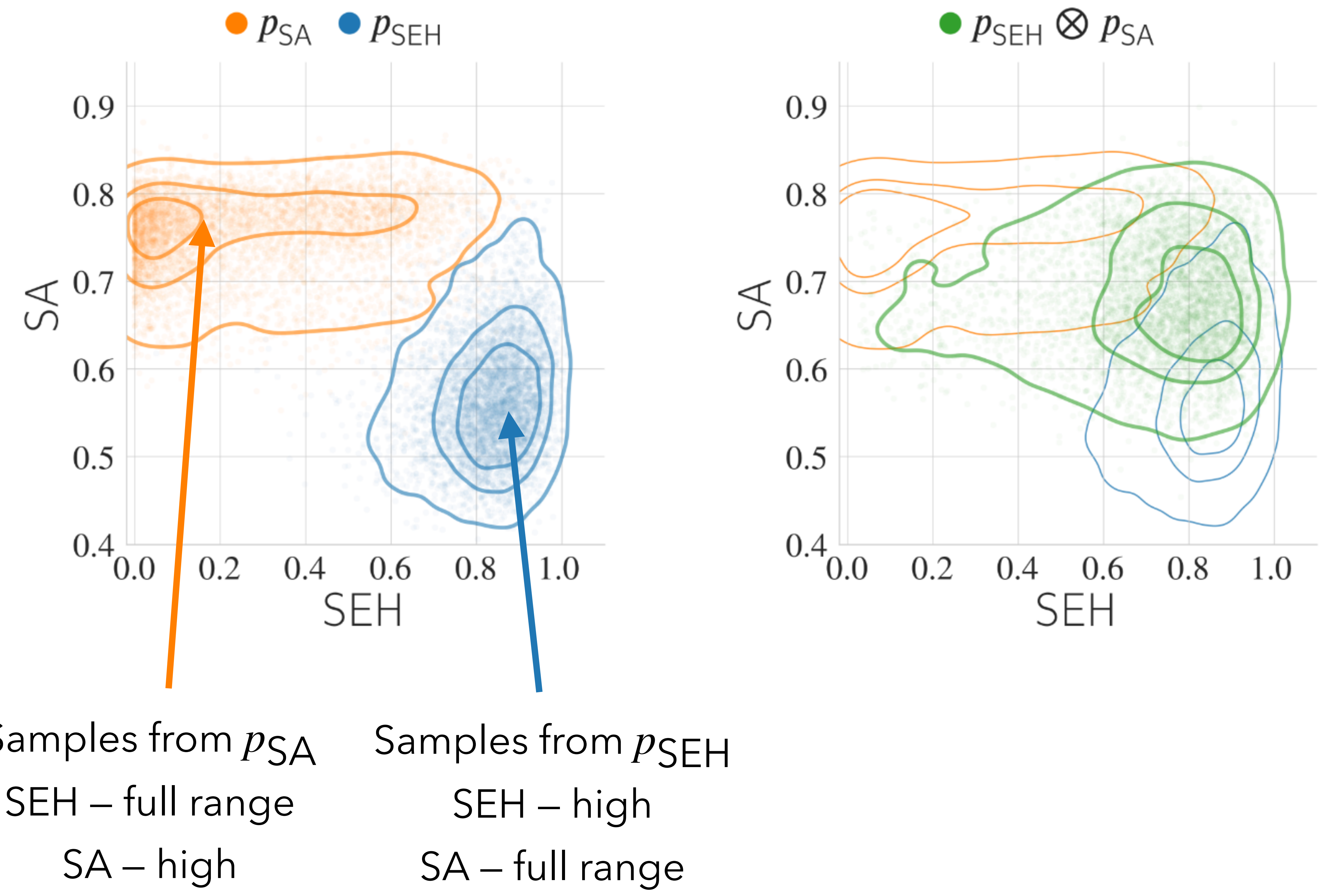
Samples from p_{SA}
SEH – full range
SA – high

Samples from p_{SEH}
SEH – high
SA – full range

Molecular property scores (“Rewards”)

- SEH – learned proxy of a protein binding score (soluble epoxide hydrolase)
- SA – synthetic availability score

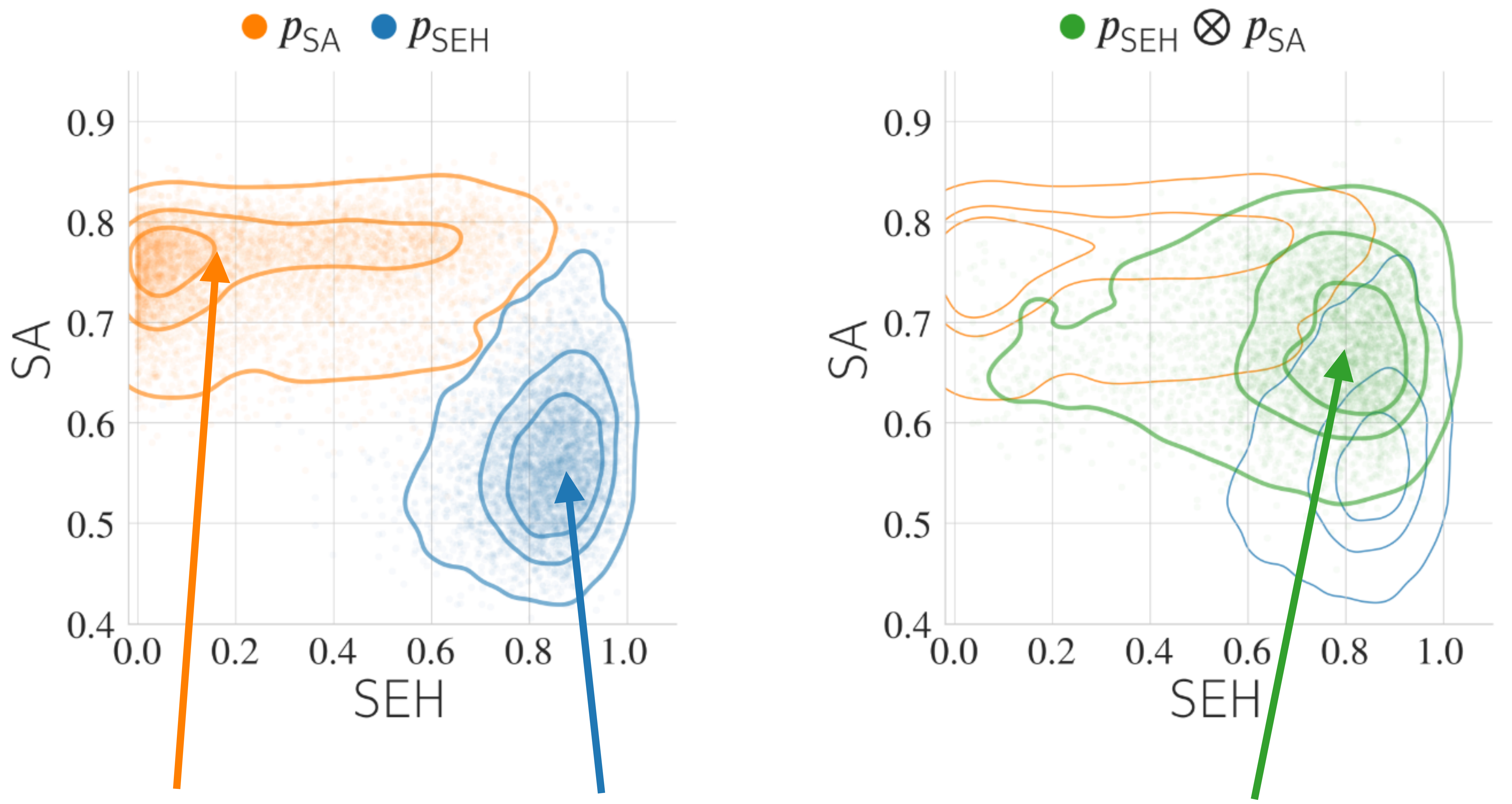
Results: GFlowNet Composition For Molecule Generation



Molecular property scores ("Rewards")

- SEH – learned proxy of a protein binding score (soluble epoxide hydrolase)
- SA – synthetic availability score

Results: GFlowNet Composition For Molecule Generation



Samples from p_{SA}
SEH – full range
SA – high

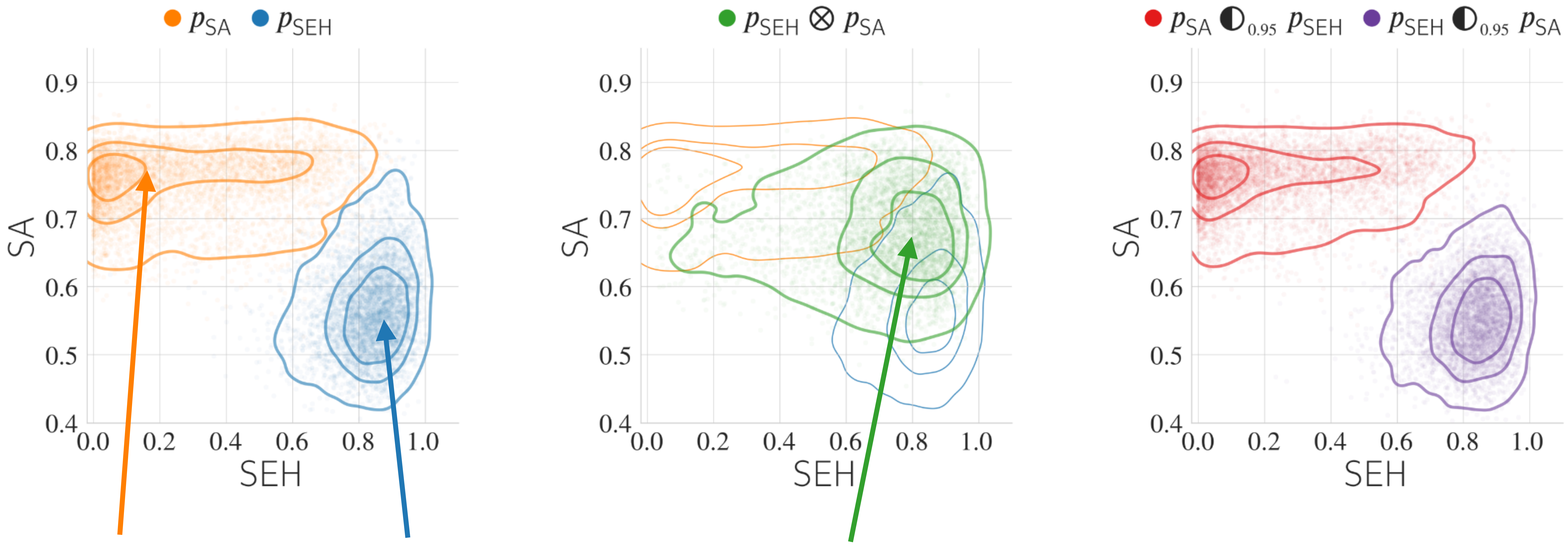
Samples from p_{SEH}
SEH – high
SA – full range

Samples from $p_{\mathrm{SEH}} \otimes p_{\mathrm{SA}}$
SA – high
SEH – high

Molecular property scores (“Rewards”)

- SEH – learned proxy of a protein binding score (soluble epoxide hydrolase)
- SA – synthetic availability score

Results: GFlowNet Composition For Molecule Generation



Samples from p_{SA}
SEH – full range
SA – high

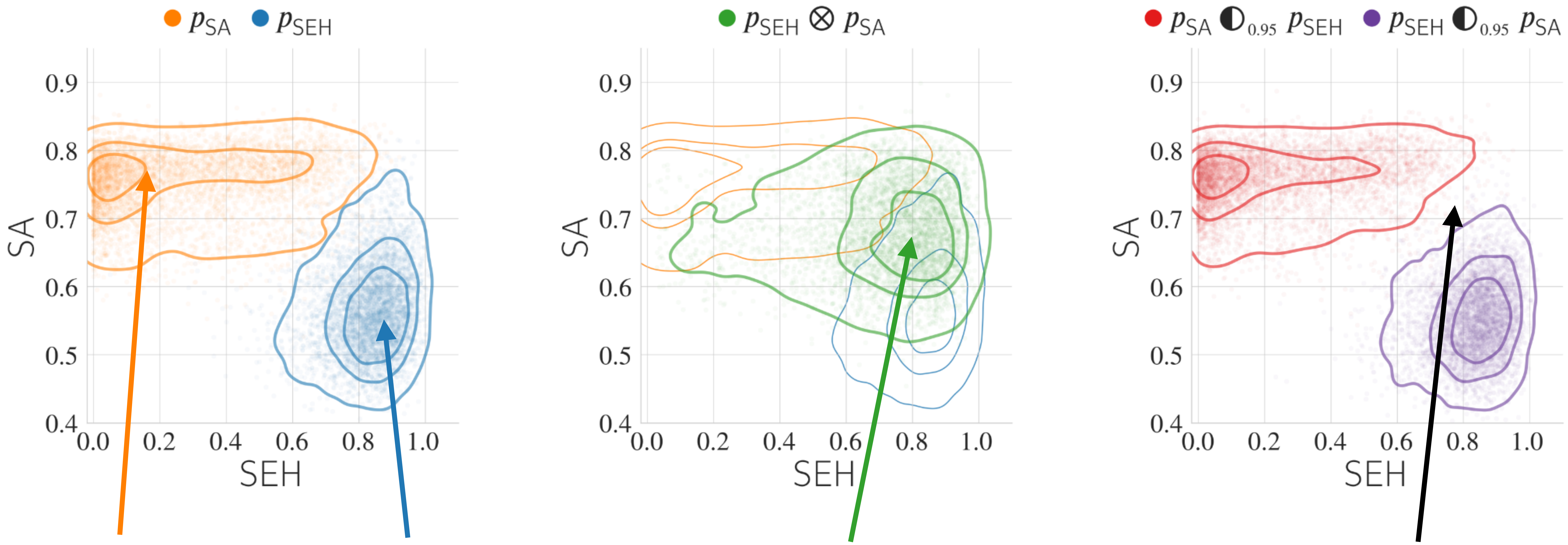
Samples from p_{SEH}
SEH – high
SA – full range

Samples from $p_{\text{SEH}} \otimes p_{\text{SA}}$
SA – high
SEH – high

Molecular property scores ("Rewards")

- SEH – learned proxy of a protein binding score (soluble epoxide hydrolase)
- SA – synthetic availability score

Results: GFlowNet Composition For Molecule Generation



Samples from p_{SA}
SEH – full range
SA – high

Samples from p_{SEH}
SEH – high
SA – full range

Samples from $p_{\text{SEH}} \otimes p_{\text{SA}}$
SA – high
SEH – high

Small density in
"shared" region

Molecular property scores ("Rewards")

- SEH – learned proxy of a protein binding score (soluble epoxide hydrolase)
- SA – synthetic availability score

Results: Toy Diffusion Composition

p_1 p_2 p_3

1) cyan digits
2) digits {“0”, “1”} (cyan or beige)
3) digits {“0”, “2”} (cyan or beige)

Results: Toy Diffusion Composition

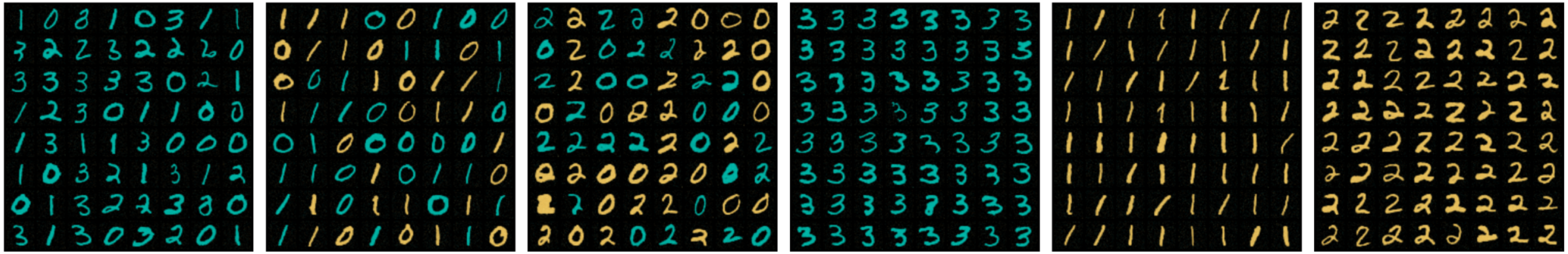
p_1 p_2 p_3

$\tilde{p}\left(x \middle| \begin{smallmatrix} y_1=1 \\ y_2=2 \end{smallmatrix}\right)$ $\tilde{p}\left(x \middle| \begin{smallmatrix} y_1=1 \\ y_2=3 \end{smallmatrix}\right)$ $\tilde{p}\left(x \middle| \begin{smallmatrix} y_1=2 \\ y_2=3 \end{smallmatrix}\right)$

1) cyan digits
2) digits {"0", "1"} (cyan or beige)
3) digits {"0", "2"} (cyan or beige)

$$\tilde{p}(x \mid y_1 = i_1, \ldots, y_n = i_n) \propto \frac{\prod_{k=1}^{n} p_{i_k}(x)}{\left(\sum_{j=1}^{m} p_j(x)\right)^{n-1}}$$

Results: Toy Diffusion Composition



p_1 p_2 p_3 $\tilde{p}\left(x \middle| {y_1=1 \atop y_2=1}\right)$ $\tilde{p}\left(x \middle| {y_1=2 \atop y_2=2}\right)$ $\tilde{p}\left(x \middle| {y_1=3 \atop y_2=3}\right)$

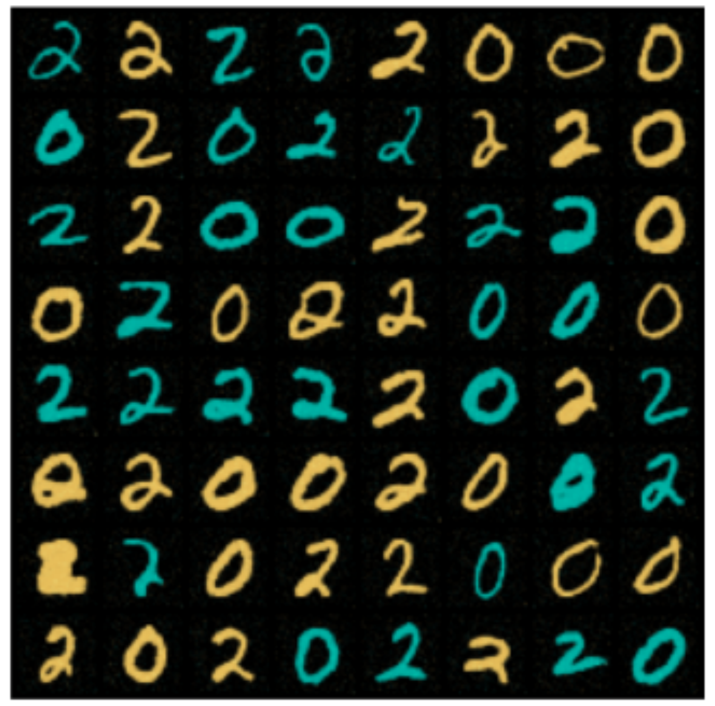
$\tilde{p}\left(x \middle| {y_1=1 \atop y_2=2}\right)$ $\tilde{p}\left(x \middle| {y_1=1 \atop y_2=3}\right)$ $\tilde{p}\left(x \middle| {y_1=2 \atop y_2=3}\right)$

$$\tilde{p}(x \mid y_1 = i_1, \ \ldots \ , y_n = i_n) \propto \frac{\prod_{k=1}^{n} p_{i_k}(x)}{\left(\sum_{j=1}^{m} p_j(x)\right)^{n-1}}$$

Results: Toy Diffusion Composition

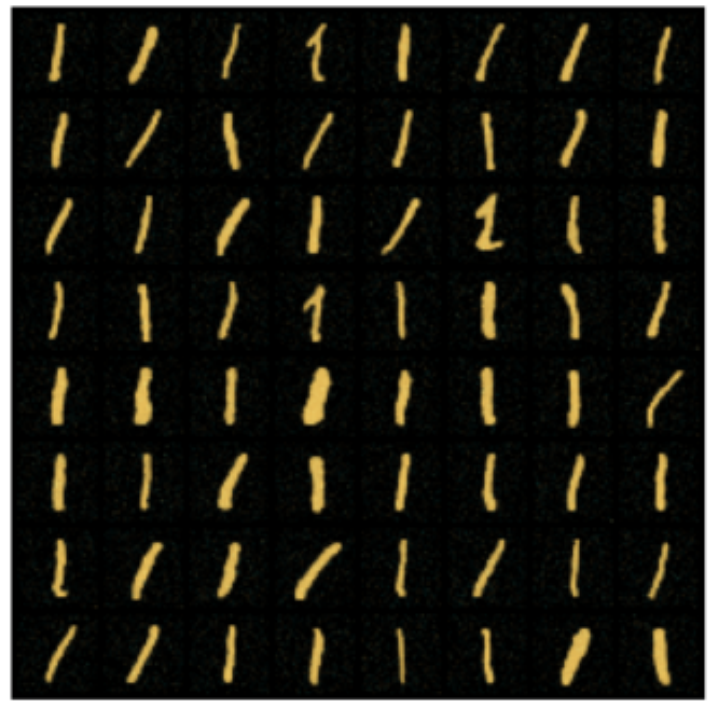
p_1

p_2



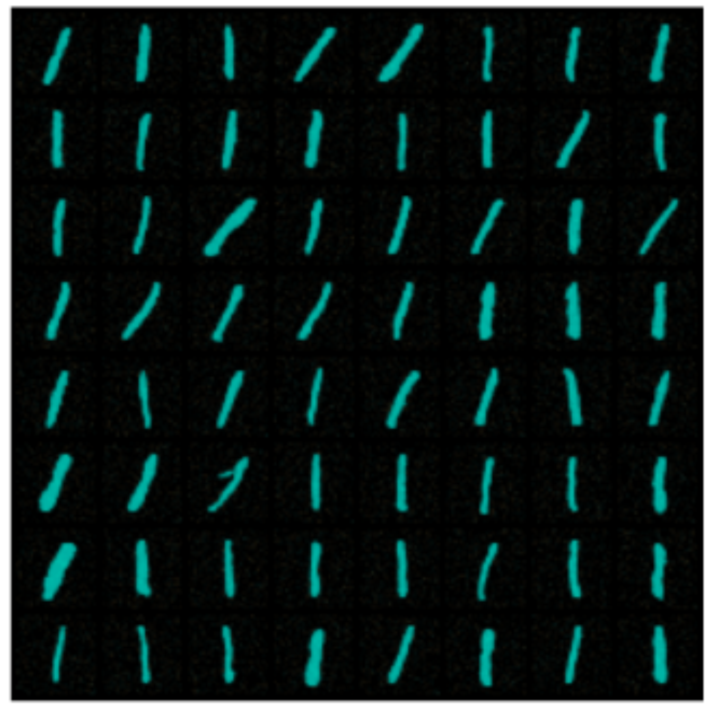
p_3

$\widetilde{p}\left(x \middle| \begin{smallmatrix} y_1=1 \\ y_2=1 \end{smallmatrix}\right)$



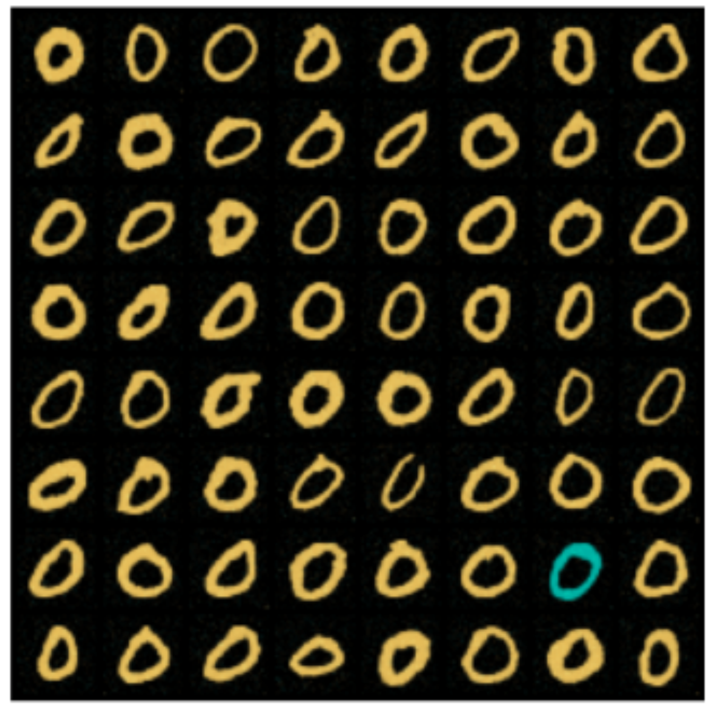
$\widetilde{p}\left(x \middle| \begin{smallmatrix} y_1=2 \\ y_2=2 \end{smallmatrix}\right)$

$\widetilde{p}\left(x \middle| \begin{smallmatrix} y_1=3 \\ y_2=3 \end{smallmatrix}\right)$



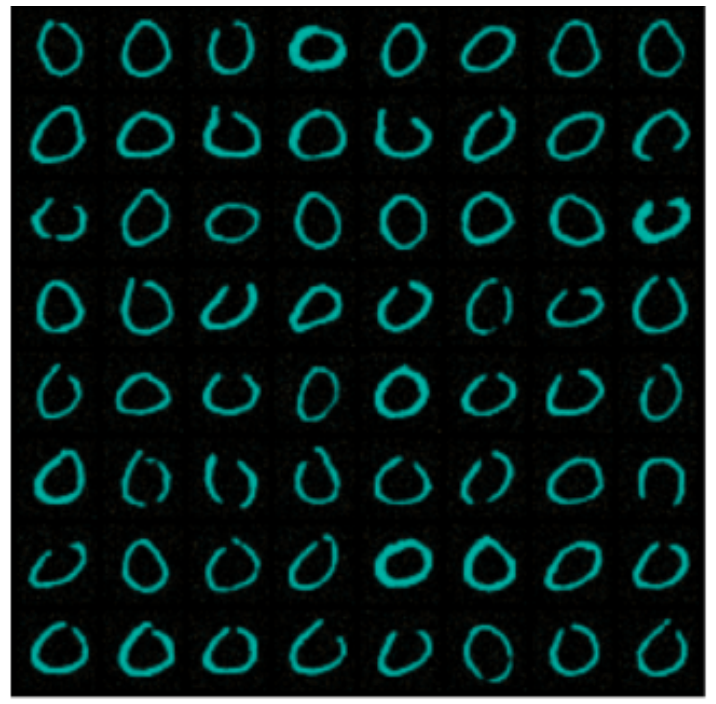
$\widetilde{p}\left(x \middle| \begin{smallmatrix} y_1=1 \\ y_2=2 \end{smallmatrix}\right)$

$\widetilde{p}\left(x \middle| \begin{smallmatrix} y_1=1 \\ y_2=3 \end{smallmatrix}\right)$



$\widetilde{p}\left(x \middle| \begin{smallmatrix} y_1=2 \\ y_2=3 \end{smallmatrix}\right)$

$$\widetilde{p}(x \mid y_1 = i_1, \ldots, y_n = i_n) \propto \frac{\prod_{k=1}^{n} p_{i_k}(x)}{\left(\sum_{j=1}^{m} p_j(x)\right)^{n-1}}$$



$\widetilde{p}\left(x \middle| \begin{smallmatrix} y_1=1 \\ y_2=2 \\ y_3=3 \end{smallmatrix}\right)$

Compositional Sculpting: Summary

New method for composition of diffusion models or GFlowNets

- **Controllable composition operations** through inference:

  Base models → (prior, observations) → posterior

  - Binary: **"Harmonic Mean"** and **"Contrast"**
  - Generalized: N-ary, parameterized, chained
- **Tractable sampling algorithm**: classifier guidance on mixture of base models

Experimental validation

- GFlowNets, controllable synthetic domain (2D grid)
- GFlowNets, molecule generation
- Diffusion models, small image generation

Thesis Overview

Contributions

- Novel principled algorithms for training and inference in deep probabilistic models
- Guarantees
    - Training: optimality of the desired target configurations
    - Inference: sampling from target distributions

Approach

- Guide complex models using signal derived from a simple auxiliary model (discriminator / coordinator)

Applications

- Image generation
- Unsupervised domain adaptation under multi-faceted distribution shift
- Multi-objective drug-like molecule generation

Models and analysis tools

- Generative adversarial networks, domain adversarial neural networks, diffusion models, GFlowNets
- Game theoretic training algorithms, optimal transport, dynamical systems, stochastic processes