Adversarial Tuning of Perturbative Parameters

in Non-Differentiable Physics Simulators

Michela Paganini

Ph.D. Student, Yale University

Affiliate, Lawrence Berkeley National Lab

with **Luke de Oliveira**, **Steve Mrenna**, **Ben Nachman**, **Chase Shimmin**, **Paul Tipton**

@WonderMicky
@mickypaganini
michela.paganini@yale.edu
mickypaganini.github.io

2nd IML Workshop

Work in Progress

Adversarial Tuning of Perturbative Parameters

in Non-Differentiable Physics Simulators

Michela Paganini

Ph.D. Student, Yale University

Affiliate, Lawrence Berkeley National Lab

@WonderMicky

@mickypaganini

michela.paganini@yale.edu

mickypaganini.github.io

with **Luke de Oliveira**, **Steve Mrenna**, **Ben Nachman**, **Chase Shimmin**, **Paul Tipton**

2nd IML Workshop

What is MC tuning?

Prior Work and (Partial) Bibliography 📖

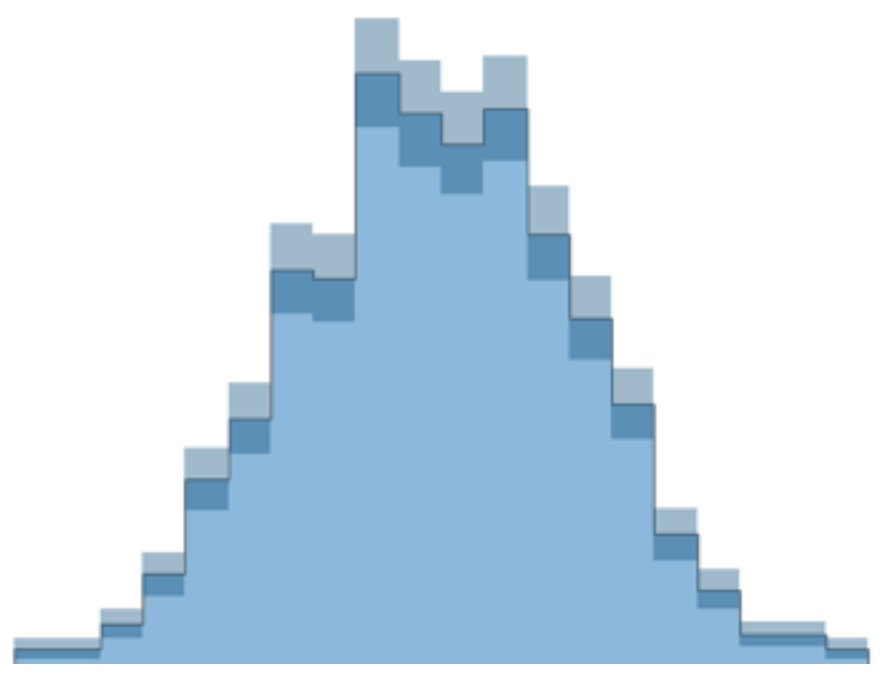
- Monte Carlo event generator validation and tuning for the LHC, arXiv:0902.4403
- Monte Carlo tuning and generator validation, arXiv:0906.0075
- Systematic event generator tuning for the LHC, arXiv:0907.2973
- Tuning Monte Carlo Generators: The Perugia Tunes, arXiv:1005.3457
- Interleaved Parton Showers and Tuning Prospects, arXiv:1011.1759
- Charged particle multiplicities in pp interactions at sqrt(s) = 0.9 and 7 TeV in a diffractive limited phase-space measured with the ATLAS detector at the LHC and new PYTHIA6 tune, ATLAS-CONF-2010-031
- New developments in event generator tuning techniques, arXiv:1005.5357
- General-purpose event generators for LHC physics, arXiv:1101.2599
- New ATLAS event generator tunes to 2010 data, ATL-PHYS-PUB-2011-008
- ATLAS tunes of PYTHIA 6 and Pythia 8 for MC11, ATL-PHYS-PUB-2011-009
- Summary of ATLAS Pythia 8 tunes, ATL-PHYS-PUB-2012-003
- Tuning of PYTHIA6 to Minimum Bias Data, *EPJ Web Conf. 60 (2013) 20056*
- QCD Monte-Carlo model tunes for the LHC, *Prog.Part.Nucl.Phys. 73 (2013) 141-187*
- Tuning PYTHIA 8.1: the Monash 2013 Tune, arXiv:1404.5630
- An Introduction to PYTHIA 8.2, arXiv:1410.3012
- Sensitivities to PDFs in parton shower MC generator reweighting and tuning, arXiv:1601.04229
- Event generator tuning using Bayesian optimization, arXiv:1610.08328
- Adversarial Variational Optimization of Non-Differentiable Simulators, arXiv:1707.07113
- Improved tuning methods for Monte Carlo generators, arXiv:1801.07187

What is MC tuning?

- MC is based on theoretical models
- There are free parameters in the theory

Constraining these parameters to take on values that are consistent with existing data is call "MC tuning".

Basic MC tuning recipe



1. **Pick set of parameter values**
2. Generate MC events
3. Plot 1D observables from MC and reference data events
4. Calculate $\chi^2(\boldsymbol{p}) = \sum_{\mathcal{O}} \sum_{b\in\mathcal{O}} w_b \frac{(f^{(b)}(\boldsymbol{p}) - \mathcal{R}_b)^2}{\Delta_b^2}$

This work: Ideas and Contributions

I. Replace **data-MC comparison** based on χ^2 agreement of 1D histograms with NN-based high-dimensional discrimination

II. Use the full radiation pattern inside jets as input, instead of a handful of observables

III. Learn event reweighting function to avoid several expensive generation calls

IV. **Tune generator parameters by back-propagation**

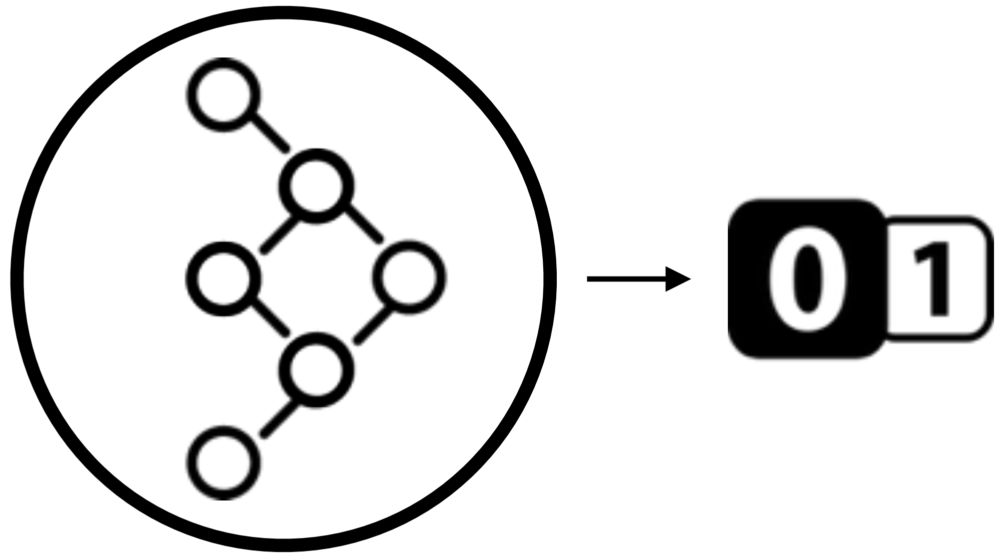
I

NN-based high-dimensional data-MC comparison.

Fundamental Approach

- Question: does $p_{\text{generator}}(x|\theta)$ match $p_{\text{data}}(x)$?

- Instead of checking 1D histograms, let a NN do this for you in high-dimensions (accounts for correlations)

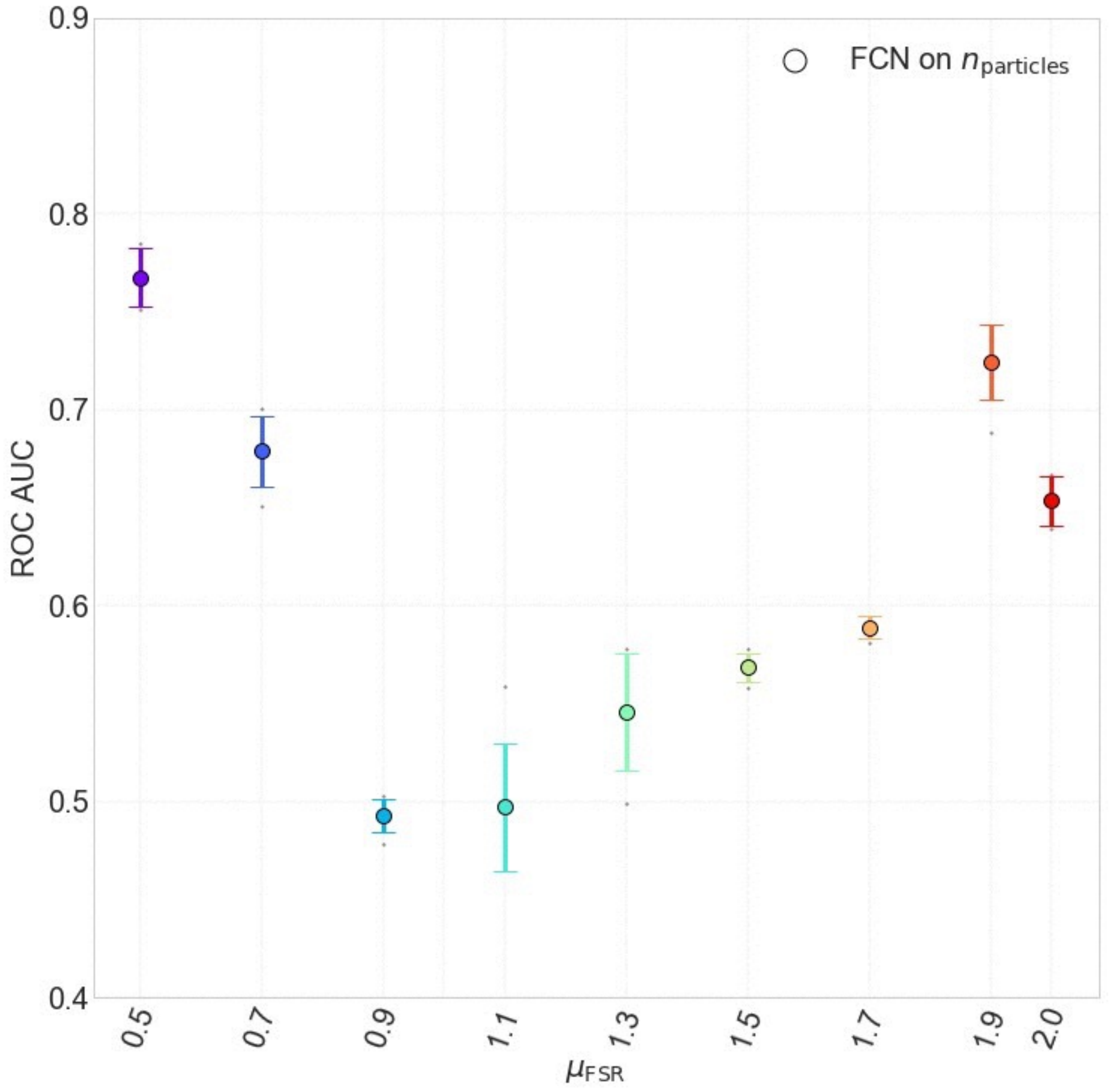


☑ Can still plot 1D distributions and calculate χ^2 to cross-check NN performance vs. your physics intuition

Experimental Setup

- Dijet events at in *pp* collisions at 13 TeV, produced with `Pythia 8.230` via the `numpythia` package in scikit-hep (500 GeV < p_T < 1000 GeV)
- Jet clustering with `FastJet` via the `pyjet` package in scikit-hep (anti-k_T R=0.4; jet p_T > 10 GeV; select 2 leading jets)
- Specify μ_{FSR} variations
- Write out to HDF5
- Train a classifier using PyTorch
- Method sensitivity in terms of binary ROC AUC

Binary Classification Tasks vs. μ_{FSR}=1.0 Baseline

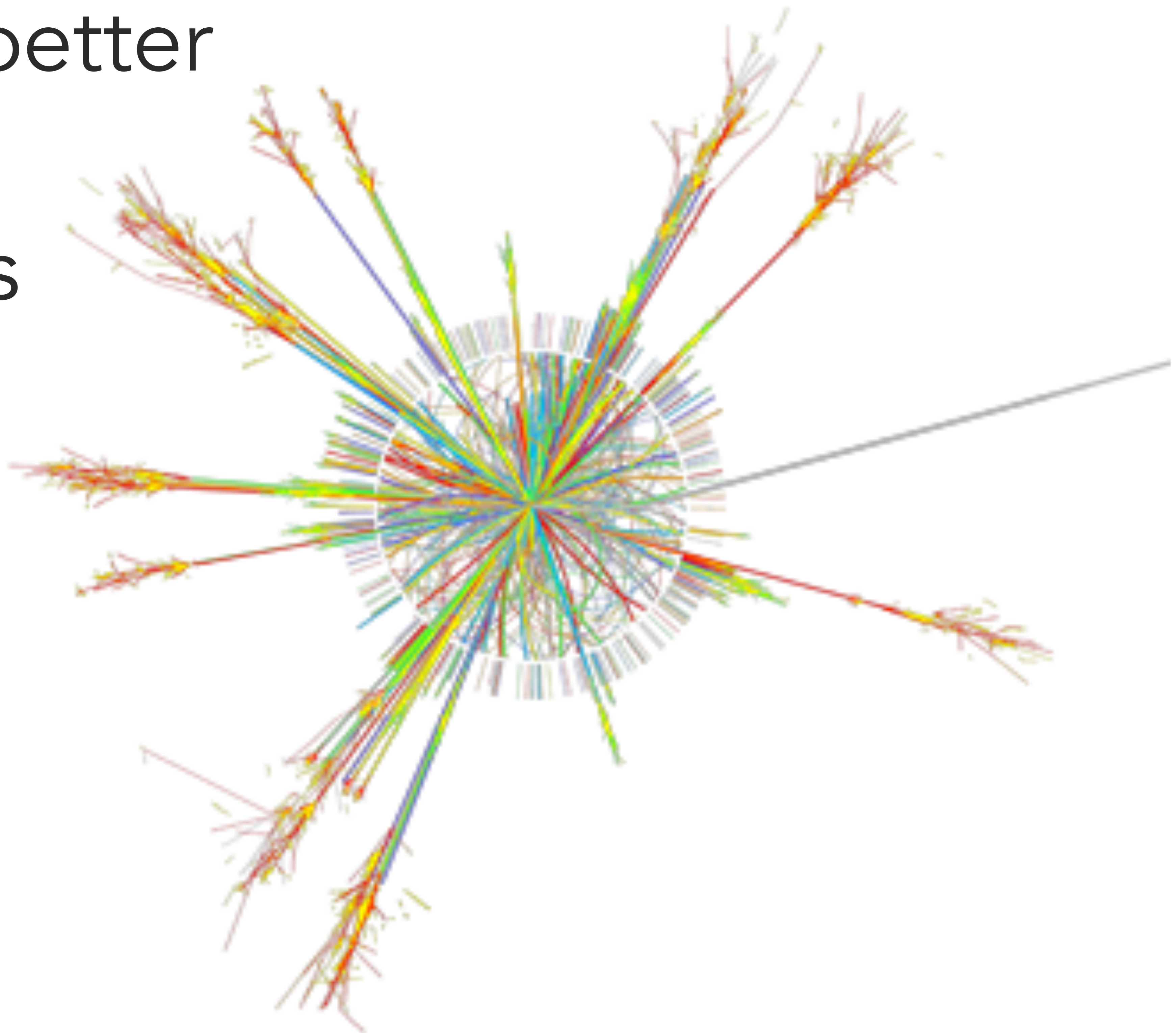


II

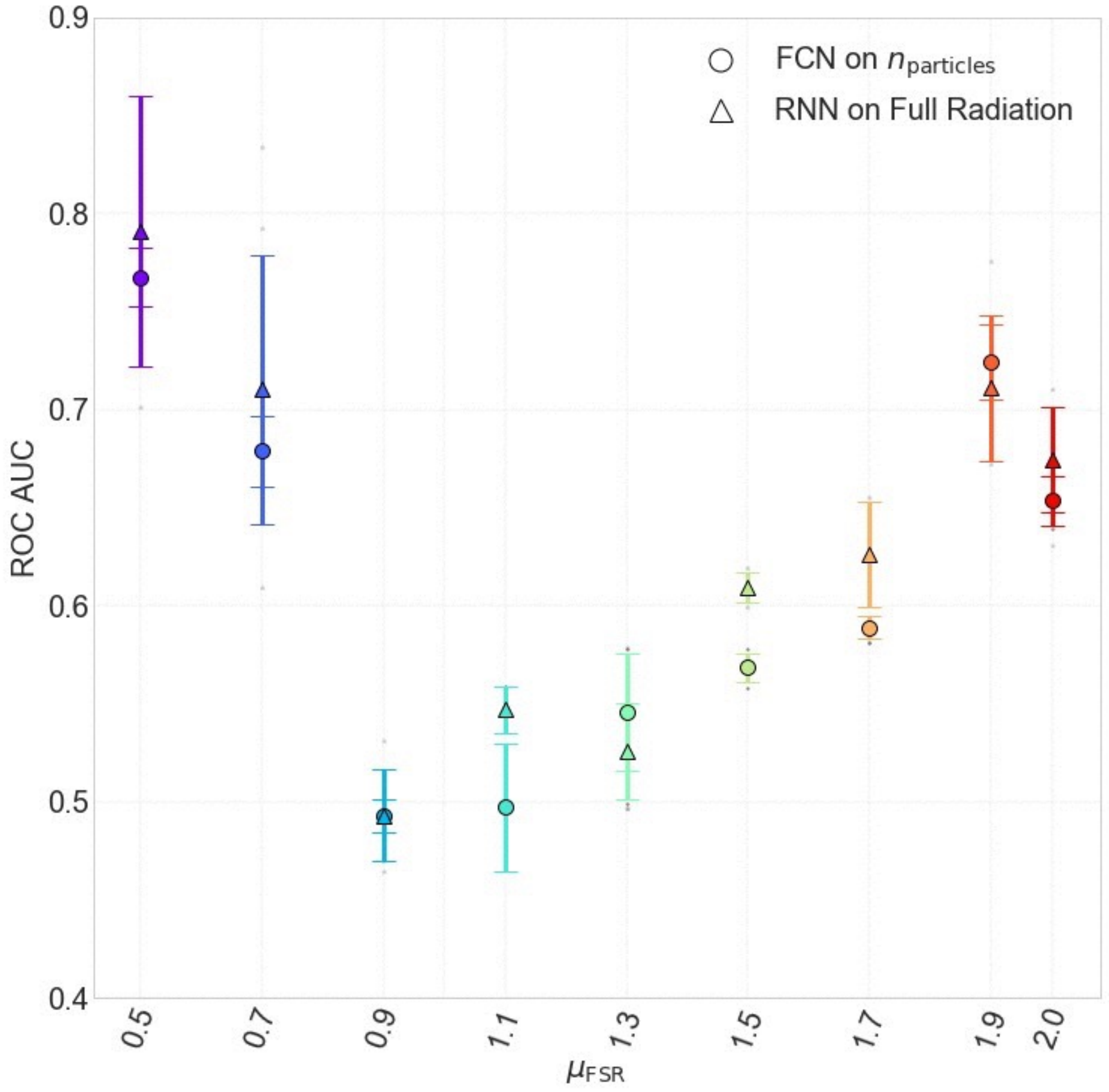
Use full radiation pattern inside jets.

Use more complex input features

- Letting a NN compare distributions scales better
- Can use much more complex inputs such as properties of all particles in the event



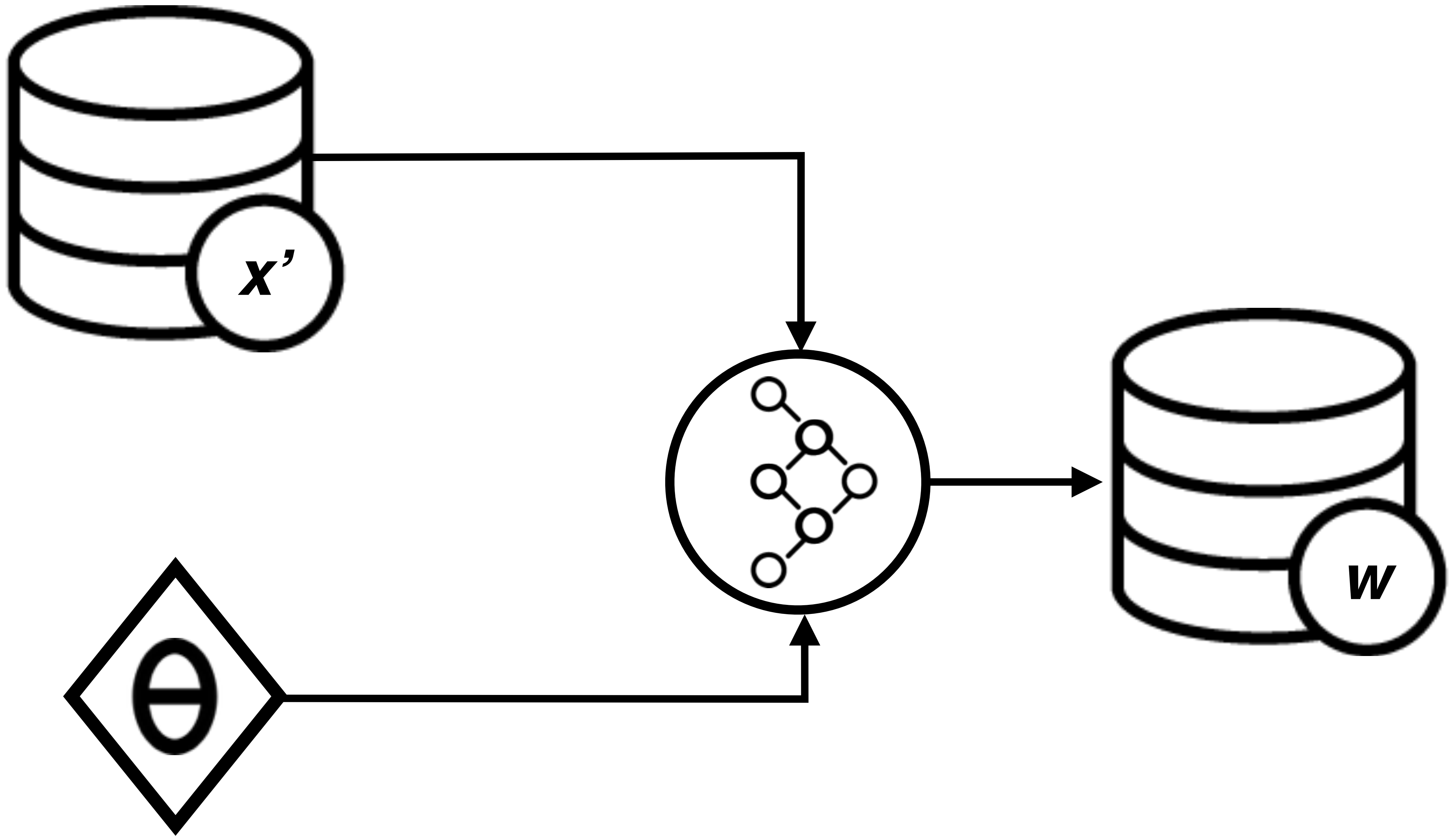
Binary Classification Tasks vs. μ_{FSR}=1.0 Baseline



III

Learn event reweighting function.

Learning the MC Event Reweighting



IV

Tune generator parameters by back-propagation.

Tuning Algorithm

Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_{D}, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{\mathrm{MC}}(\theta_0)$, the pre-computed empirical distribution of simulated events.

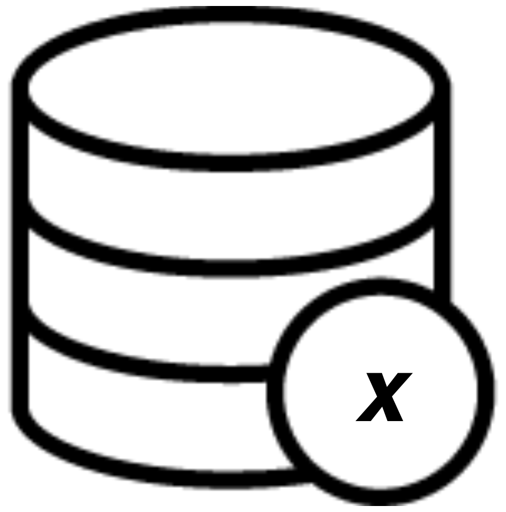
```
while θ has not converged do
    for t = 0, ..., n_D do
        Sample {x^(i)}_{i=1}^b ~ P_data a batch of events from the real data.
        Sample {x'^(i)}_{i=1}^b ~ P_MC(θ_0) a batch of simulated events.
        Compute {w^(i)(θ)}_{i=1}^b = {f(x'^(i)|θ)}_{i=1}^b simulated batch event weights.
        g_Ω ← ∇_Ω [ 1/(b + Σ_i w^(i)(θ)) Σ_{i=1}^b ( L(D(x^(i)), 1) + w^(i)(θ) L(D(x'^(i)), 0) ) ]
        Ω ← Ω − Opt(Ω, g_Ω; λ)
    end for
    Sample {x'^(i)}_{i=1}^b ~ P_MC(θ_0) a batch of simulated events.
    Compute {w^(i)(θ)}_{i=1}^b = {f(x'^(i)|θ)}_{i=1}^b simulated batch event weights.
    g_θ ← ∇_θ 1/(Σ_i w^(i)(θ)) Σ_{i=1}^b w^(i)(θ) L(D(x'^(i)), 1)
    θ ← θ − Opt(θ, g_θ; λ)
end while
```

Where:

$$g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta) \mathscr{L}\left(D(x'^{(i)}), 0\right) \right) \right]$$

$$g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta) \mathscr{L}\left(D(x'^{(i)}), 1\right)$$

Tuning Algorithm



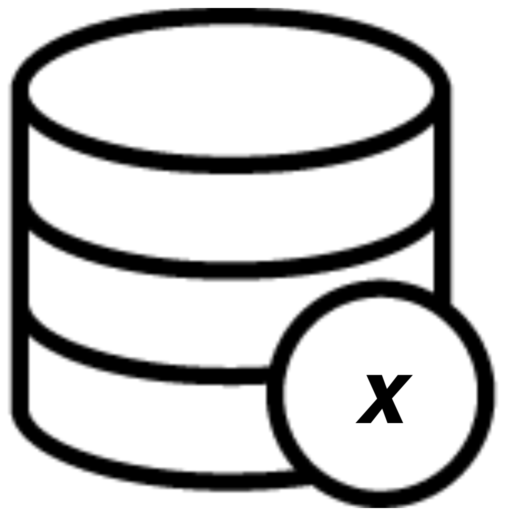
Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_D, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{MC}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
  for $t = 0, ..., n_D$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{data}$ a batch of events from the real data.
    Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{MC}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
    $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 0\right)\right)\right]$
    $\Omega \leftarrow \Omega - \mathsf{Opt}(\Omega, g_\Omega; \lambda)$
  end for
  Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{MC}(\theta_0)$ a batch of simulated events.
  Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
  $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 1\right)$
  $\theta \leftarrow \theta - \mathsf{Opt}(\theta, g_\theta; \lambda)$
end while

Tuning Algorithm

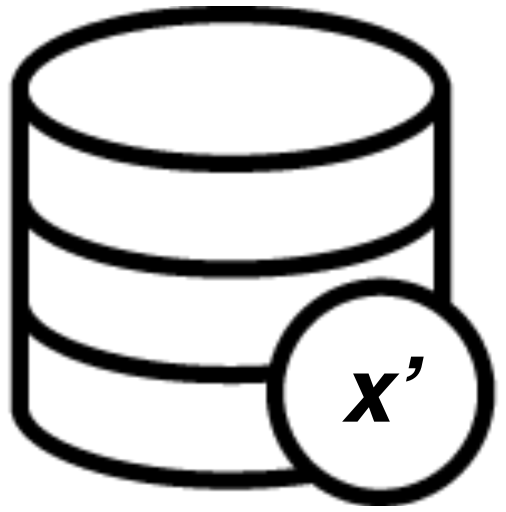


Algorithm 1 Proposed tuning algorithm.

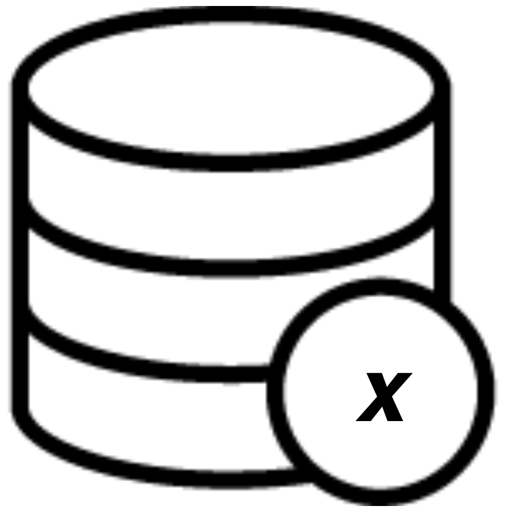
Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_D, the number of training iterations of the discriminator per generator iteration. $\mathcal{L}$, the binary cross-entropy loss function.

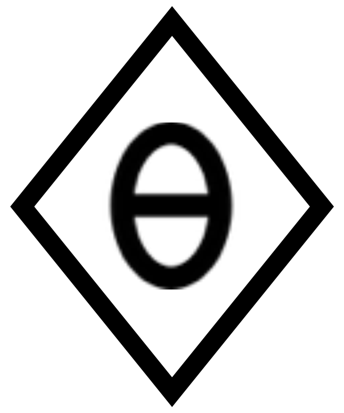
Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{MC}(\theta_0)$, the pre-computed empirical distribution of simulated events.

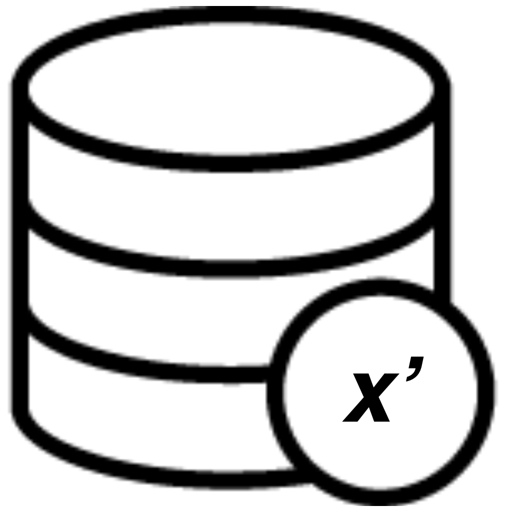
while θ has not converged **do**
  for $t = 0, ..., n_D$ **do**
    Sample $\{x^{(i)}\}_{i=1}^b \sim \mathbb{P}_{data}$ a batch of events from the real data.
    Sample $\{x'^{(i)}\}_{i=1}^b \sim \mathbb{P}_{MC}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^b = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^b$ simulated batch event weights.
    $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^b \left(\mathcal{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta)\mathcal{L}\left(D(x'^{(i)}), 0\right)\right)\right]$
    $\Omega \leftarrow \Omega - \mathsf{Opt}(\Omega, g_\Omega; \lambda)$
  end for
  Sample $\{x'^{(i)}\}_{i=1}^b \sim \mathbb{P}_{MC}(\theta_0)$ a batch of simulated events.
  Compute $\{w^{(i)}(\theta)\}_{i=1}^b = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^b$ simulated batch event weights.
  $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^b w^{(i)}(\theta)\mathcal{L}\left(D(x'^{(i)}), 1\right)$
  $\theta \leftarrow \theta - \mathsf{Opt}(\theta, g_\theta; \lambda)$
end while



Tuning Algorithm







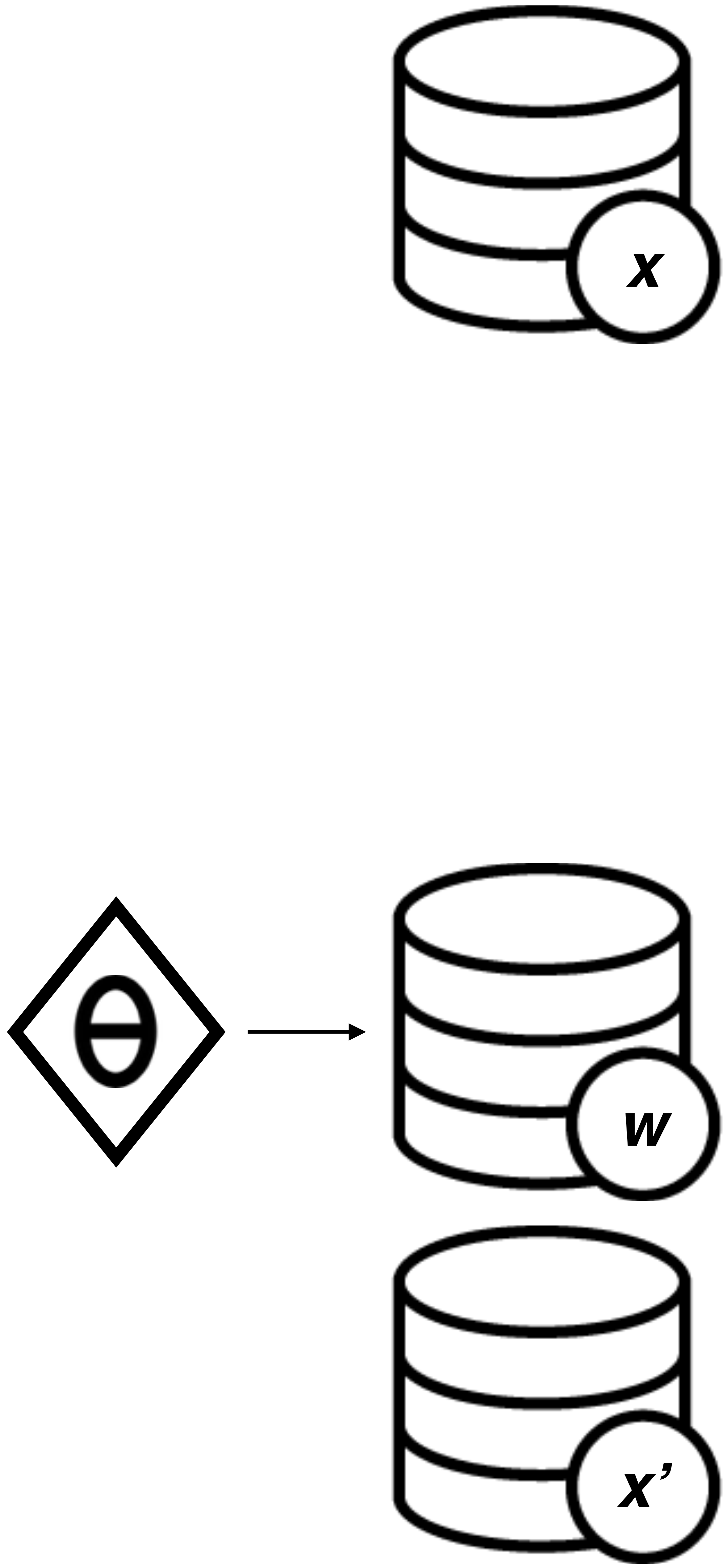
Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_D, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{MC}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
  for $t = 0, ..., n_D$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{data}$ a batch of events from the real data.
    Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{MC}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
    $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}),1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}),0\right)\right)\right]$
    $\Omega \leftarrow \Omega - \mathsf{Opt}(\Omega, g_\Omega; \lambda)$
  end for
  Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{MC}(\theta_0)$ a batch of simulated events.
  Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
  $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}),1\right)$
  $\theta \leftarrow \theta - \mathsf{Opt}(\theta, g_\theta; \lambda)$
end while

Tuning Algorithm



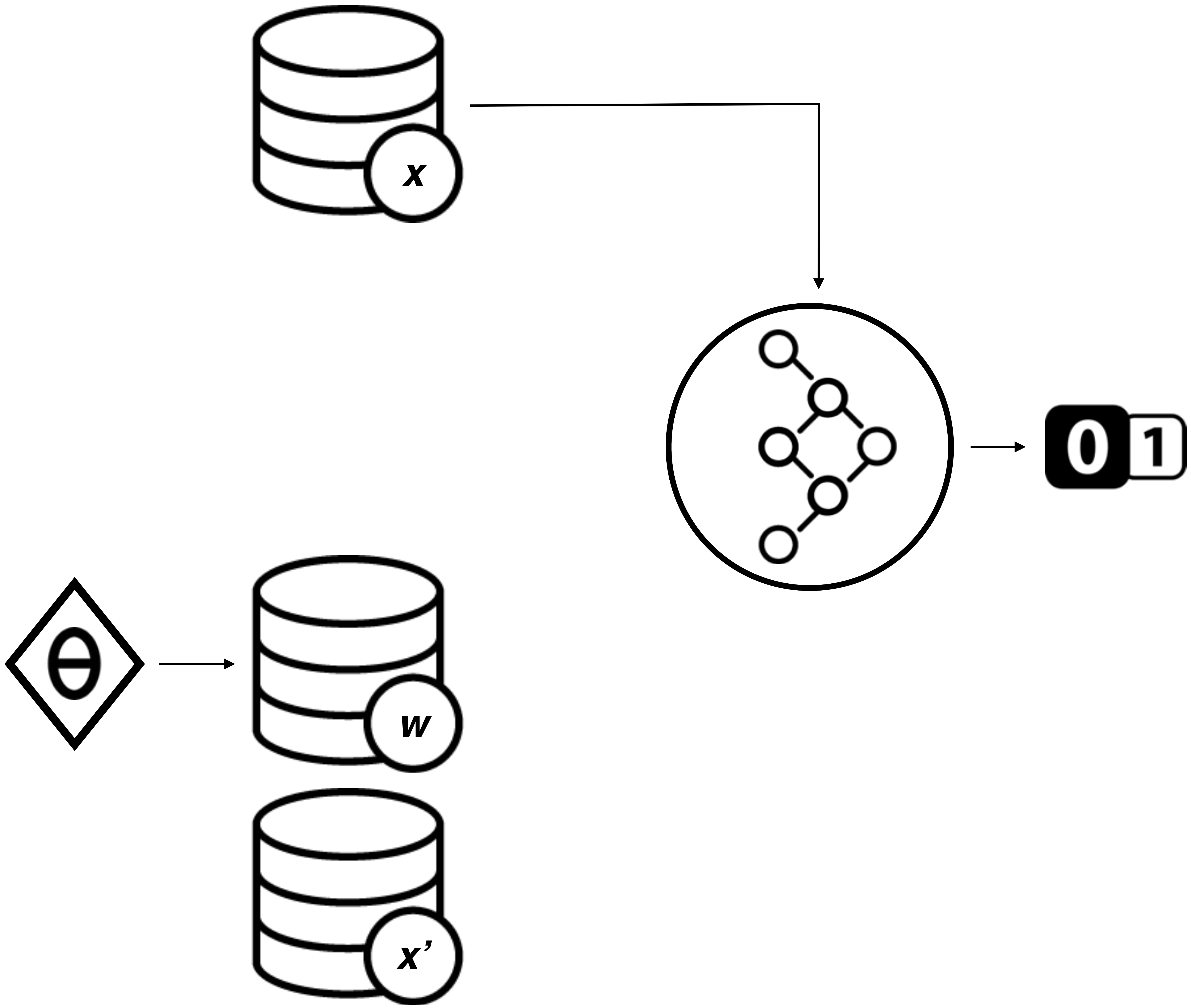
Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_{D}, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{\mathrm{MC}}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
    for $t = 0, ..., n_{\mathrm{D}}$ **do**
        Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{data}}$ a batch of events from the real data.
        Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
        Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
        $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 0\right)\right)\right]$
        $\Omega \leftarrow \Omega - \mathsf{Opt}(\Omega, g_\Omega; \lambda)$
    end for
    Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
    $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 1\right)$
    $\theta \leftarrow \theta - \mathsf{Opt}(\theta, g_\theta; \lambda)$
end while

Tuning Algorithm



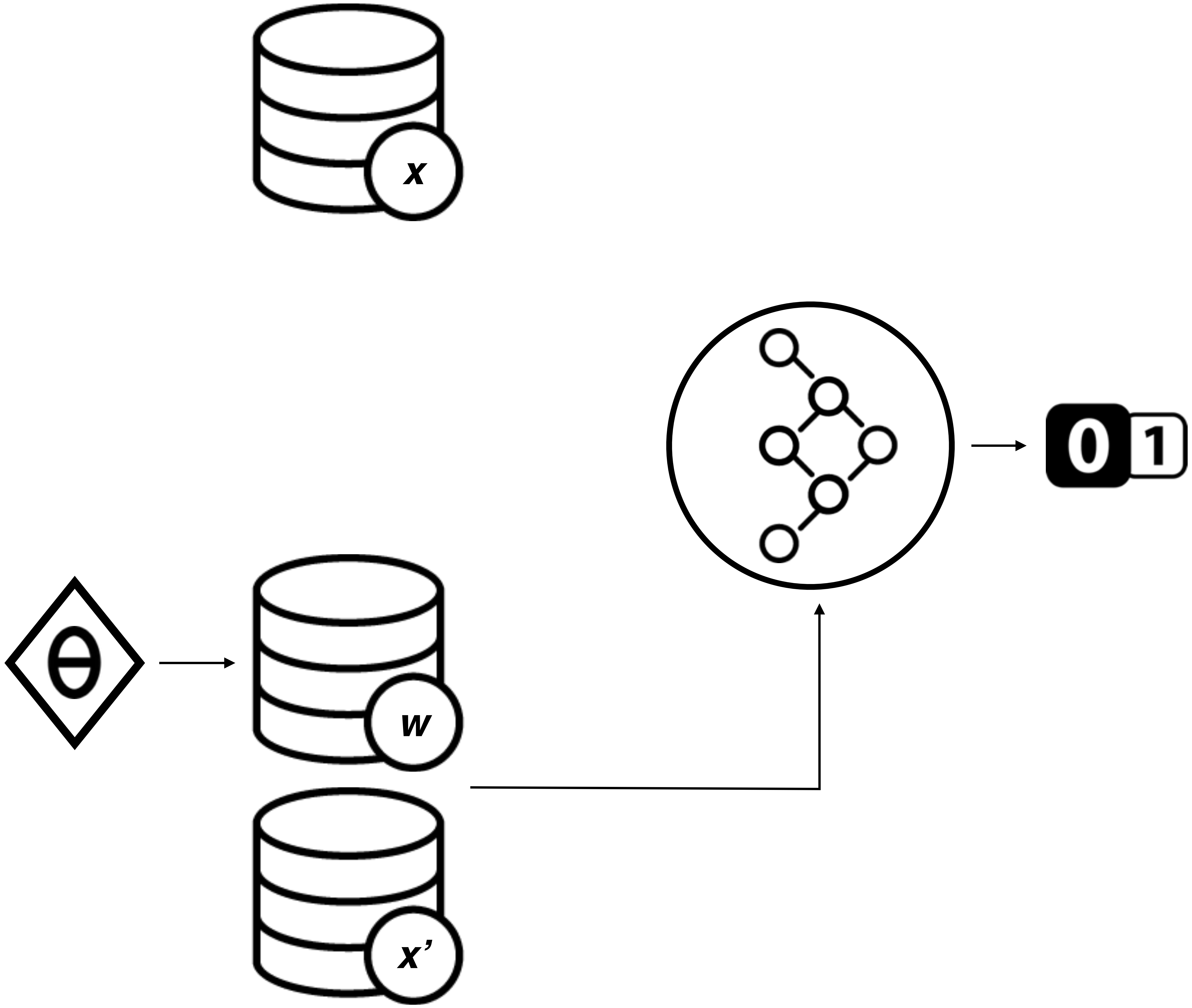
Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_{D}, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{\mathrm{MC}}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
    for $t = 0, ..., n_{\mathrm{D}}$ **do**
        Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{data}}$ a batch of events from the real data.
        Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
        Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
        $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 0\right)\right)\right]$
        $\Omega \leftarrow \Omega - \mathsf{Opt}(\Omega, g_\Omega; \lambda)$
    end for
    Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
    $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 1\right)$
    $\theta \leftarrow \theta - \mathsf{Opt}(\theta, g_\theta; \lambda)$
end while

Tuning Algorithm



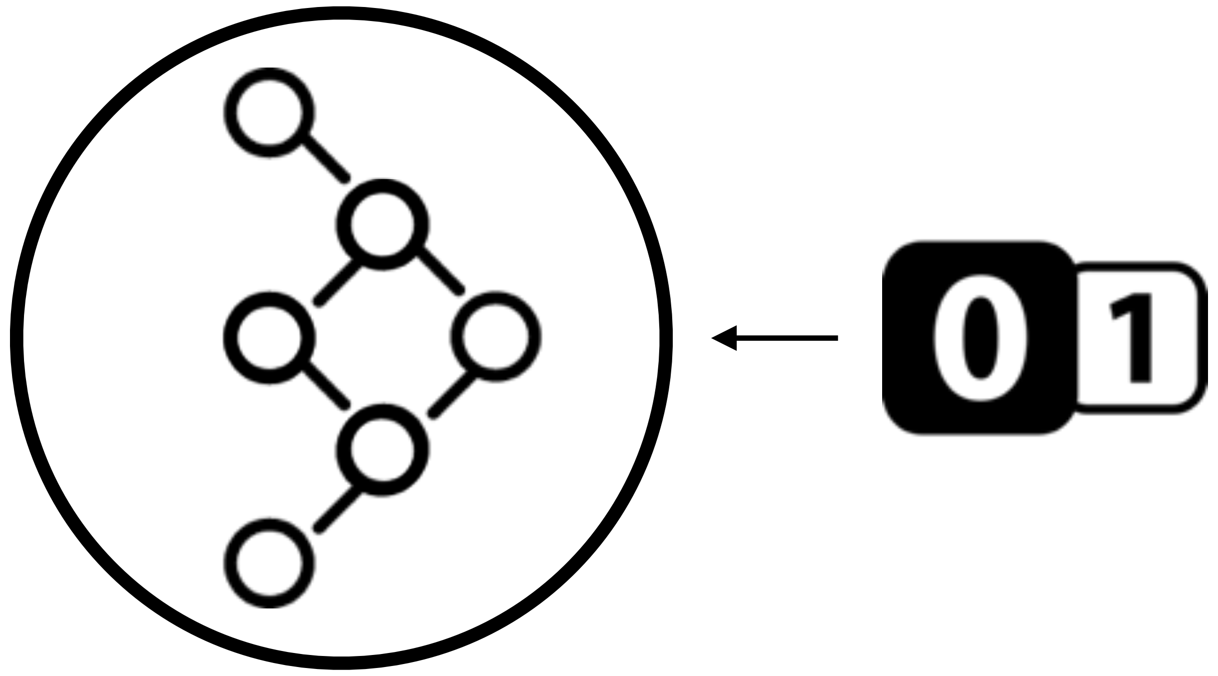
Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_{D}, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{\mathrm{MC}}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
  for $t = 0, ..., n_{\mathrm{D}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{data}}$ a batch of events from the real data.
    Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
    $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 0\right)\right)\right]$
    $\Omega \leftarrow \Omega - \mathrm{Opt}(\Omega, g_\Omega; \lambda)$
  end for
  Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
  Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
  $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 1\right)$
  $\theta \leftarrow \theta - \mathrm{Opt}(\theta, g_\theta; \lambda)$
end while

Tuning Algorithm



Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_{D}, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{\mathrm{MC}}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
- **for** $t = 0, ..., n_{\mathrm{D}}$ **do**
  - Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{data}}$ a batch of events from the real data.
  - Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
  - Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
  - $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}),1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}),0\right)\right)\right]$
  - $\Omega \leftarrow \Omega - \mathsf{Opt}(\Omega, g_\Omega; \lambda)$
- **end for**
- Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
- Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
- $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}),1\right)$
- $\theta \leftarrow \theta - \mathsf{Opt}(\theta, g_\theta; \lambda)$

end while

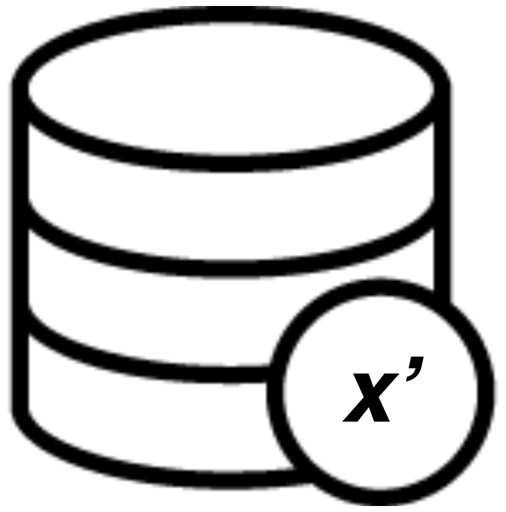
Tuning Algorithm

Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_{D}, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{\mathrm{MC}}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
  for $t = 0, ..., n_{\mathrm{D}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{data}}$ a batch of events from the real data.
    Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
    $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 0\right)\right)\right]$
    $\Omega \leftarrow \Omega - \mathsf{Opt}(\Omega, g_\Omega; \lambda)$
  end for
  Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
  Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
  $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 1\right)$
  $\theta \leftarrow \theta - \mathsf{Opt}(\theta, g_\theta; \lambda)$
end while



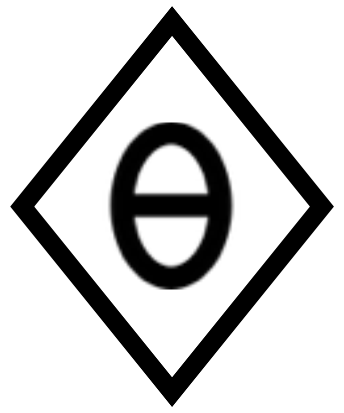
Tuning Algorithm

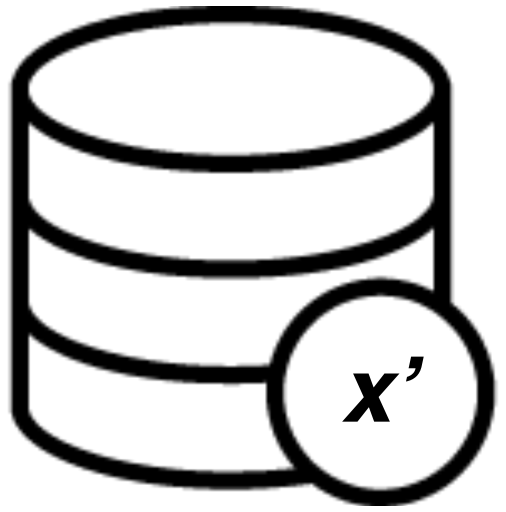
Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_{D}, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{\mathrm{MC}}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
  for $t = 0, ..., n_{\mathrm{D}}$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{data}}$ a batch of events from the real data.
    Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
    $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 0\right)\right)\right]$
    $\Omega \leftarrow \Omega - \mathsf{Opt}(\Omega, g_\Omega; \lambda)$
  end for
  Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
  Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
  $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 1\right)$
  $\theta \leftarrow \theta - \mathsf{Opt}(\theta, g_\theta; \lambda)$
end while





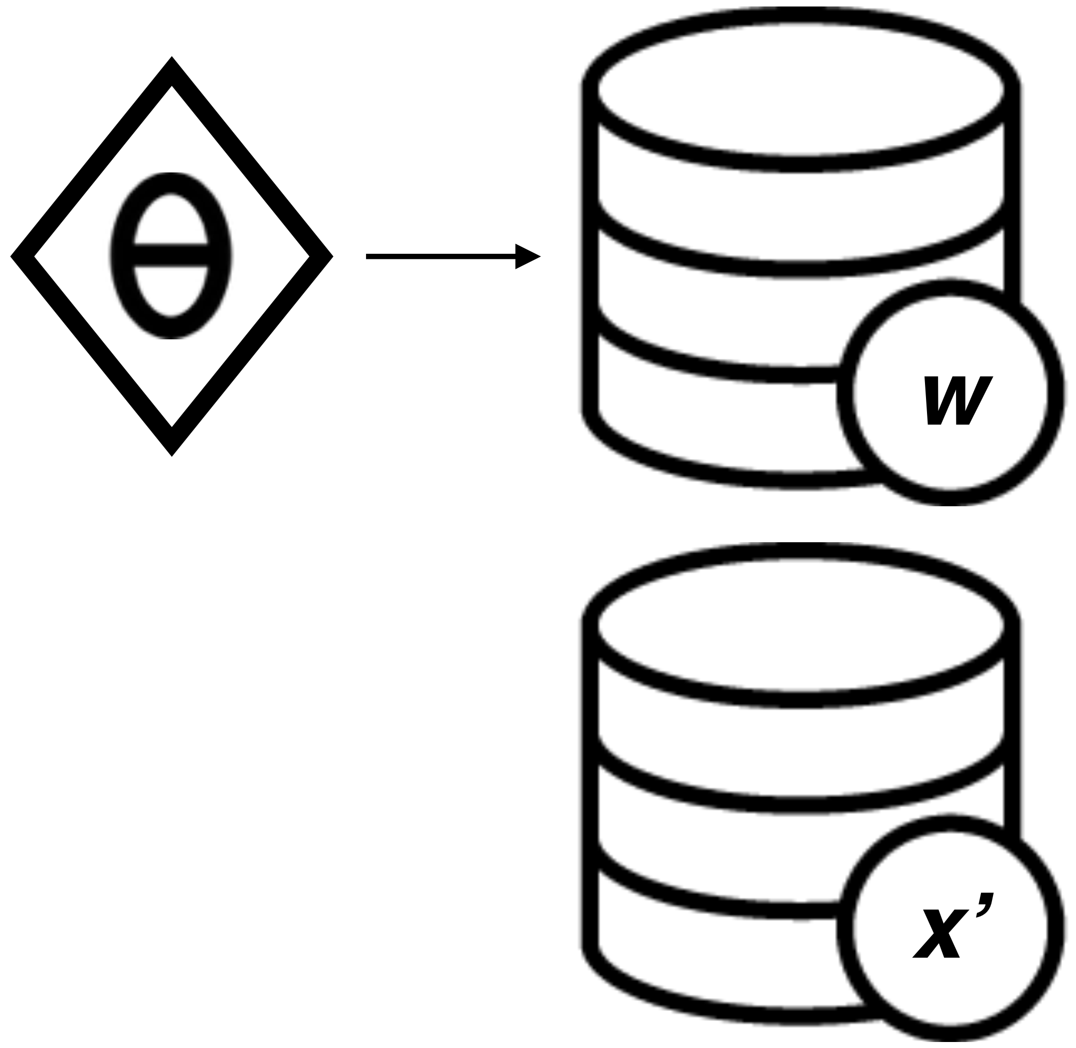
Tuning Algorithm

Algorithm 1 Proposed tuning algorithm.

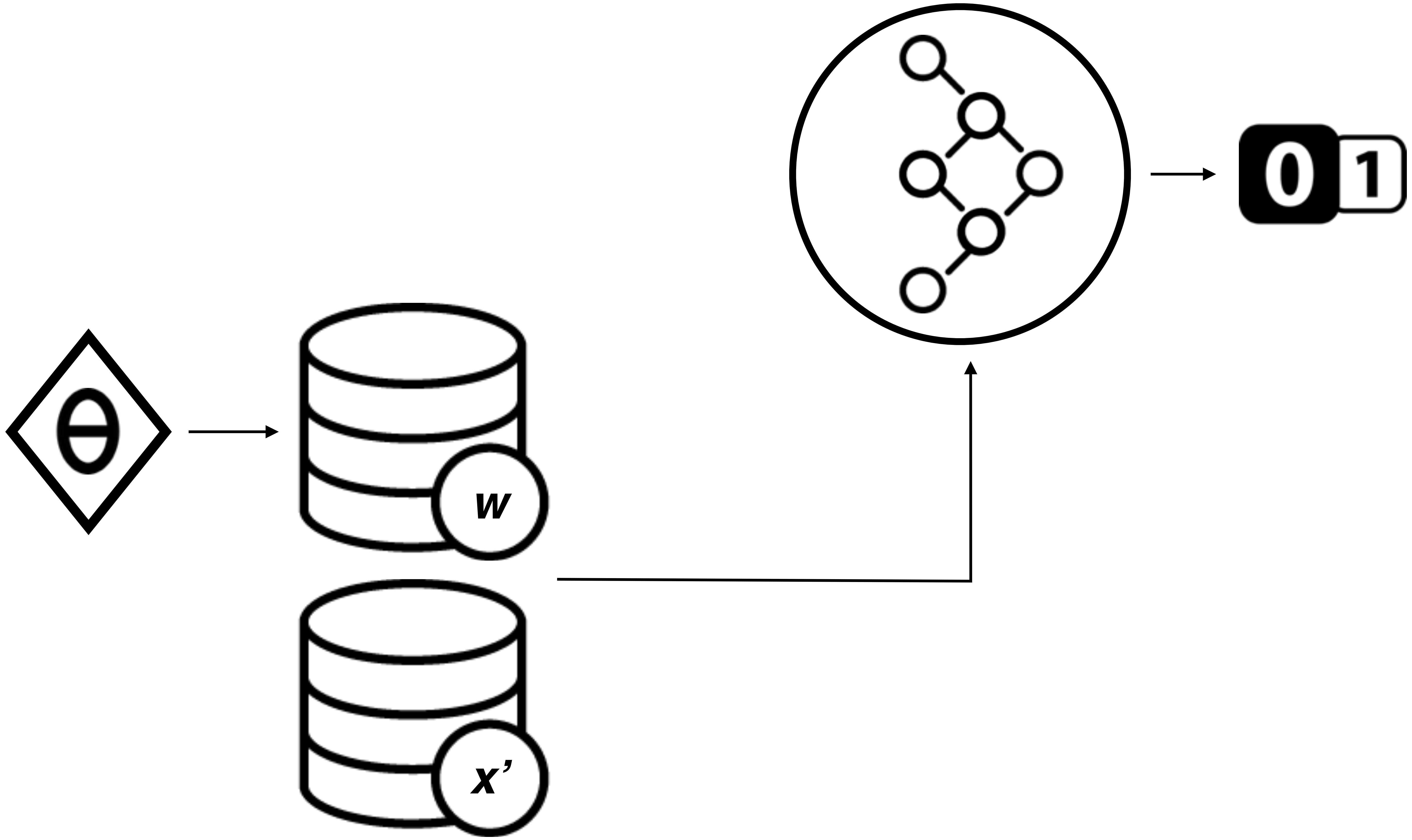
Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_{D}, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{\mathrm{MC}}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
    for $t = 0, ..., n_{\mathrm{D}}$ **do**
        Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{data}}$ a batch of events from the real data.
        Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
        Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
        $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 0\right)\right)\right]$
        $\Omega \leftarrow \Omega - \mathrm{Opt}(\Omega, g_\Omega; \lambda)$
    end for
    Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\mathrm{MC}}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
    $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 1\right)$
    $\theta \leftarrow \theta - \mathrm{Opt}(\theta, g_\theta; \lambda)$
end while



Tuning Algorithm



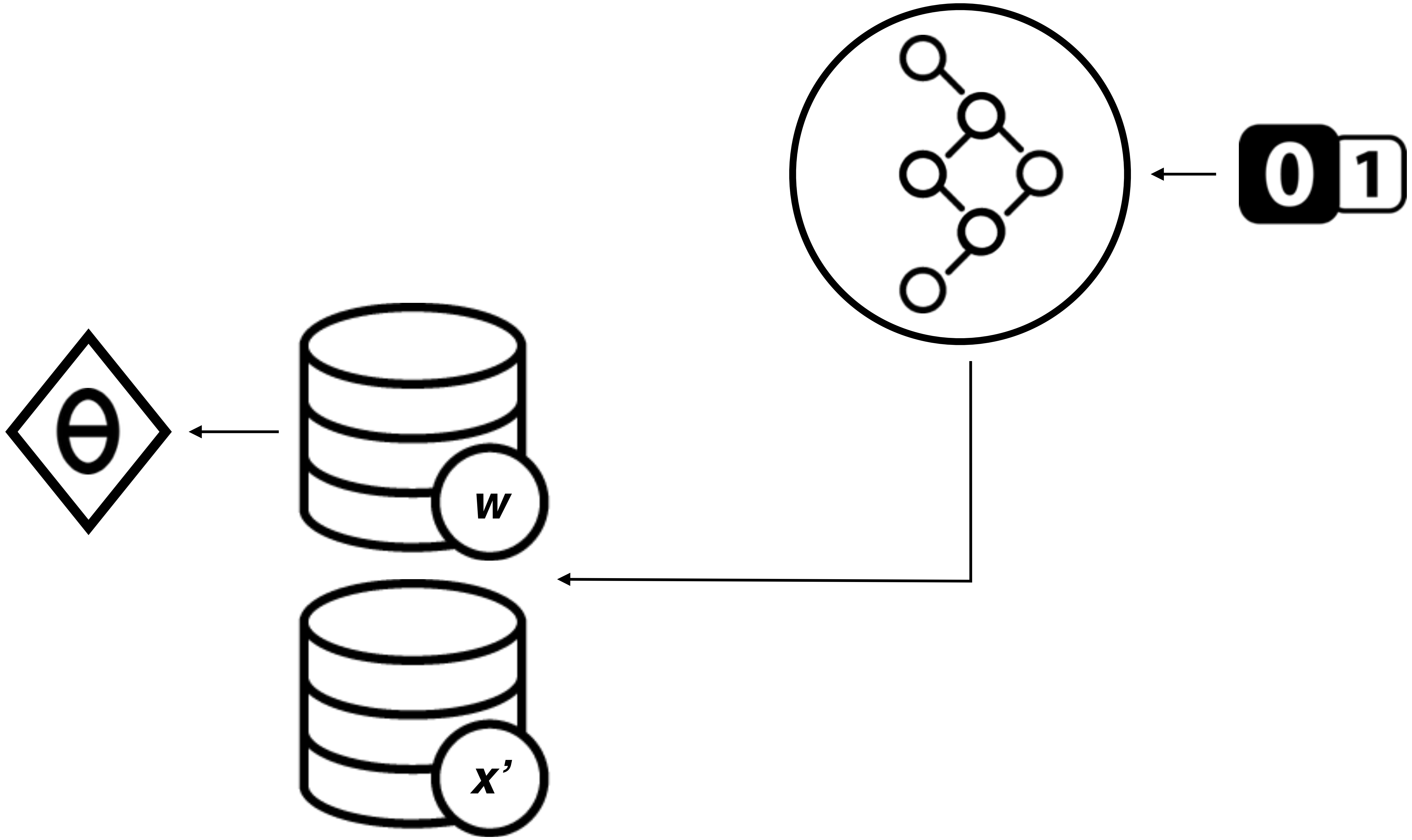
Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_{D}, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{\mathrm{MC}}(\theta_0)$, the pre-computed empirical distribution of simulated events.

```
while θ has not converged do
    for t = 0, ..., n_D do
        Sample {x^(i)}_{i=1}^b ~ P_data a batch of events from the real data.
        Sample {x'^(i)}_{i=1}^b ~ P_MC(θ_0) a batch of simulated events.
        Compute {w^(i)(θ)}_{i=1}^b = {f(x'^(i)|θ)}_{i=1}^b simulated batch event weights.
        g_Ω ← ∇_Ω [ 1/(b + Σ_i w^(i)(θ)) Σ_{i=1}^b ( L(D(x^(i)), 1) + w^(i)(θ) L(D(x'^(i)), 0) ) ]
        Ω ← Ω − Opt(Ω, g_Ω; λ)
    end for
    Sample {x'^(i)}_{i=1}^b ~ P_MC(θ_0) a batch of simulated events.
    Compute {w^(i)(θ)}_{i=1}^b = {f(x'^(i)|θ)}_{i=1}^b simulated batch event weights.
    g_θ ← ∇_θ 1/(Σ_i w^(i)(θ)) Σ_{i=1}^b w^(i)(θ) L(D(x'^(i)), 1)
    θ ← θ − Opt(θ, g_θ; λ)
end while
```

Tuning Algorithm



Algorithm 1 Proposed tuning algorithm.

Require: : Opt, an optimization algorithm (e.g., RMSProp) with hyperparameters λ. b, the batch size. n_D, the number of training iterations of the discriminator per generator iteration. $\mathscr{L}$, the binary cross-entropy loss function.

Require: : Ω_0, initial discriminator's parameters. θ_0, initial value of the perturbative parameters to tune. $\mathbb{P}_{MC}(\theta_0)$, the pre-computed empirical distribution of simulated events.

while θ has not converged **do**
  for $t = 0, ..., n_D$ **do**
    Sample $\{x^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{\text{data}}$ a batch of events from the real data.
    Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{MC}(\theta_0)$ a batch of simulated events.
    Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
    $g_\Omega \leftarrow \nabla_\Omega \left[\frac{1}{b+\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} \left(\mathscr{L}\left(D(x^{(i)}), 1\right) + w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 0\right)\right)\right]$
    $\Omega \leftarrow \Omega - \mathsf{Opt}(\Omega, g_\Omega; \lambda)$
  end for
  Sample $\{x'^{(i)}\}_{i=1}^{b} \sim \mathbb{P}_{MC}(\theta_0)$ a batch of simulated events.
  Compute $\{w^{(i)}(\theta)\}_{i=1}^{b} = \{f\left(x'^{(i)}|\theta\right)\}_{i=1}^{b}$ simulated batch event weights.
  $g_\theta \leftarrow \nabla_\theta \frac{1}{\sum_i w^{(i)}(\theta)} \sum_{i=1}^{b} w^{(i)}(\theta)\mathscr{L}\left(D(x'^{(i)}), 1\right)$
  $\theta \leftarrow \theta - \mathsf{Opt}(\theta, g_\theta; \lambda)$
end while

Summary: Ideas and Contributions

I. Replace **data-MC comparison** based on χ^2 agreement of 1D histograms with NN-based high-dimensional discrimination

II. Use the full radiation pattern inside jets as input, instead of a handful of observables

III. Learn event reweighting function to avoid several expensive generation calls

IV. **Tune generator parameters by back-propagation**

Thank you!

Questions?

You can find me at: michela.paganini@yale.edu