entropy

MDPI

Article

Fast Model Selection and Hyperparameter Tuning for Generative Models

Luming Chen * and Sujit K. Ghosh

Department of Statistics, North Carolina State University, Raleigh, NC 27695, USA; sujit.ghosh@ncsu.edu
* Correspondence: lchen21@ncsu.edu

Abstract: Generative models have gained significant attention in recent years. They are increasingly used to estimate the underlying structure of high-dimensional data and artificially generate various kinds of data similar to those from the real world. The performance of generative models depends critically on a good set of hyperparameters. Yet, finding the right hyperparameter configuration can be an extremely time-consuming task. In this paper, we focus on speeding up the hyperparameter search through adaptive resource allocation, early stopping underperforming candidates quickly and allocating more computational resources to promising ones by comparing their intermediate performance. The hyperparameter search is formulated as a non-stochastic best-arm identification problem where resources like iterations or training time constrained by some predetermined budget are allocated to different hyperparameter configurations. A procedure which uses hypothesis testing coupled with Successive Halving is proposed to make the resource allocation and early stopping decisions and compares the intermediate performance of generative models by their exponentially weighted Maximum Means Discrepancy (MMD). The experimental results show that the proposed method selects hyperparameter configurations that lead to a significant improvement in the model performance compared to Successive Halving for a wide range of budgets across several real-world applications.

Keywords: integral probability metric; hypothesis testing; Maximum Mean Discrepancy; multi-armed bandits; generative adversarial networks

Citation: Chen, L.; Ghosh, S.K. Fast Model Selection and Hyperparameter Tuning for Generative Models. *Entropy* **2024**, *26*, 150. https://doi.org/10.3390/e26020150

Academic Editor: Donald J. Jacobs

Received: 8 January 2024
Revised: 1 February 2024
Accepted: 6 February 2024
Published: 9 February 2024

1. Introduction

The performance of the generative models depends heavily on so-called hyperparameters which include the model architecture, the choice of training objective, regularization and training algorithms. However, the choice of these hyperparameters is often problem-dependent, and it is unknown a priori which configuration would produce the best results in terms of a specific distance or divergence measure. With the rich set of objective functions and training algorithms proposed in recent years and the growing number of tuning parameters associated with them, it is crucial to develop computationally efficient search methods for hyperparameter configurations that yield models with a desired performance within a fixed budget constraint.

The problem of efficient model search and hyperparameter optimization has recently been dominated by Bayesian optimization approaches, e.g., [1–3], which speed up the search for good configurations by modeling the underlying structure of the search space. These approaches select and evaluate hyperparameter configurations in an adaptive manner, trying to find good configurations faster than baselines such as random search or grid search. While Bayesian optimization is efficient in tuning few hyperparameters, its efficiency often degrades significantly when the search dimension becomes much higher. Wang et al. [4] showed that for high-dimensional problems, standard Bayesian optimization methods perform similarly to random search. Moreover, traditional Bayesian optimization methods (that are often based on Gaussian processes) can only work on continuous hyperparameters, but not categorical ones (e.g., the choice of the training objective). The vast

majority of these hyperparameter selection procedures consider the training of machine learning models to be black-box procedures, and only evaluate models after they have been trained to convergence. It thus seems natural to ask the following question: *Can we terminate some of these poor-performing hyperparameter settings early to speed up hyperparameter optimization?* This is one of the primary questions that we address in this work.

In fact, there is a line of research that perceives hyperparameter optimization as an adaptive computational resource allocation problem, where the type of resources can be iterations, execution time, data samples, or even total money to spend with a cloud computing provider. These approaches evaluate partially trained models and make decisions on the fly, allocating more resources to promising hyperparameter configurations while early stopping those that are not. They allow for the training of multiple models simultaneously. And by quickly eliminating unpromising ones and paralleling the model training for different hyperparameter configurations, more hyperparameter configurations can be examined. Swersky et al. [5], Domhan et al. [6] and Klein et al. [7] made parametric assumptions on the convergence behavior of learning curves to devise early stopping rules. However, these assumptions are strong and restrictive for the kinds of learning curves that are typically found in training machine learning models. In contrast, Sparks et al. [8] cast it as a multi-armed bandit problem, viewing each hyperparameter configuration as an 'arm', and resources constrained by some predetermined budget are allocated among them by some heuristic rule. Jamieson and Talwalkar [9] and Li et al. [10] studied a similar problem but in the non-stochastic setting and based their resource allocation strategies on the Successive Halving algorithm originally proposed in Karnin et al. [11] for stochastic settings.

However, most existing methodologies on fast model selection and hyperparameter tuning, including all of those discussed above, mainly focus on supervised learning tasks where the data are labeled and the performance of the partially trained models is represented by their losses on a hold-out test set. However, for unsupervised learning tasks, such as many generative modeling tasks that are performed to generate high-quality samples, e.g., [12], fast model search approaches, though important, have been largely scarce in the literature. In light of the rapidly growing literature on generative modeling, it is important to be able to perform efficient model search and hyperparameter tuning based on the quality of samples generated from the candidate models.

In this work, we focus on the best hyperparameter configuration identification problem for generative models. We perceive it as an adaptive resource allocation problem with a given budget (e.g., iteration and training time). We build our approach upon the non-stochastic multi-armed bandit formulation proposed by Jamieson and Talwalkar [9], which makes minimal assumptions on the convergence behavior of the model performance during the training process. Note that, in this work, we restrict the term "Generative Models" to represent models that learn the underlying probability distribution in input data to generate similar samples. To evaluate the performance of generative models, we compute a sample-based distance metric between the samples generated from partially trained models and those from the reference distribution. Particularly, we base our evaluation criterion on the Maximum Mean Discrepancy (MMD) [13] that measures the closeness of the generated samples to the reference distribution. By incorporating statistical tests between partially trained models into the evaluation process, our method effectively identifies the poor-performing configurations early on and allocates more resources to promising configurations.

The remainder of this paper is organized as follows. In Section 2.1, we review the non-stochastic best-arm identification problem and the Successive Halving algorithm. In Sections 2.2 and 2.3, we present the proposed *Adaptive Successive Halving algorithm* (**AdaptSH**) and provide intuition for it through an example. In Section 3, we present empirical results comparing AdaptSH with Successive Halving. We conclude with a discussion in Section 4.

2. Methods

In this section, we present the AdaptSH algorithm. We start with a brief review of the non-stochastic best-arm identification problem and Successive Halving. We subsequently introduce our choice of the metric for comparing the intermediate performance of generative models to decide which models should be trained further, and we provide intuition for our choice via a simple example. We then introduce our proposed statistical test and how we incorporate it into Successive Halving to help us distinguish between candidate models and make early stopping decisions.

2.1. Non-Stochastic Best-Arm Identification and Successive Halving

The non-stochastic best-arm identification problem, originally proposed in Jamieson and Talwalkar [9], considers a very general setting that encompasses the hyperparameter optimization problem of interest. It only assumes that the sequence of the losses of each arm (hyperparameter configuration) eventually converges without making any assumptions on the rate of the convergence, monotonicity, or smoothness of the sequence. Hence, it is generally applicable to a wide variety of problems including minimizing a non-convex objective using stochastic gradient descent or some other iterative algorithms. Let K denote the total number of arms and let $\ell_{k,j}$ denote the validation error of the kth arm after training for j units of resources (e.g., iterations). For all $k \in \{1, 2, \ldots, K\}$, assume $\nu_k = \lim_{j\to\infty} \ell_{k,j}$ exists. The goal is to identify $\arg\min_k \nu_k$ when the resources are constrained by some predetermined budget. Successive Halving, shown in Algorithm 1, is proposed by Jamieson and Talwalkar [9] to solve the above problem. The strategy of Successive Halving follows its name: given a set of K arms and a budget B, it splits the given budget evenly across $\log_2(K)$ elimination rounds, uniformly allocates the resources to remaining arms at each round, evaluates their intermediate performance, throws out the worst half until one arm remains. By the design of the algorithm, it allocates exponentially more resources to more promising configurations.

Algorithm 1 Successive Halving

Input: Budget B, K models $\mathbf{M}_1, \ldots, \mathbf{M}_K$

$S_0 = \{1, 2, \ldots, K\}$
Initialize $i = 0$
$n = K$
while $B > 0$ **and** $n >= 2$ **do**
  Allocate $r_i = \left\lfloor \frac{B}{n\lceil \log_2(n) \rceil} \right\rfloor$ units of resource to each model in S_i
  $R_i = \sum_{j=0}^{i} r_j$
  Sort the intermediate losses of the models in S_i such that $\ell_{\sigma_i(1),R_i} \leq \ell_{\sigma_i(2),R_i} \leq \cdots \leq \ell_{\sigma_i(n),R_i}$, where $\sigma_i(\cdot)$ is a bijection from {1, 2, . . . , n} to S_i
  $S_{i+1} = \{\sigma_i(j) \mid 1 \leq j \leq \lfloor \frac{n}{2} \rfloor\}$
  $B = B - nr_i$
  $n = \lfloor \frac{n}{2} \rfloor$
  $i = i + 1$
end while

2.2. Exponentially Weighed Average of MMD^2

While in Successive Halving, half of the configurations are discarded at each elimination round, it is not entirely clear why we should do so. Indeed, it is not clear what proportion to discard in each round would lead to better results without prior knowledge about the convergence behavior of the sequences of losses. We propose to use statistical tests to detect when two models have separated in their performance to make on-the-fly elimination decisions. Before we jump into the statistical test, we first introduce our choice of metric to represent the intermediate performance of generative models, based on which we develop our statistical test to distinguish between their model performance.

While for supervised learning tasks, models are usually compared by their validation errors on a hold-out set, there is no such straightforward measure for generative model comparisons. Although there are a number of evaluation measures for generative models that have been proposed in recent years including the average log-likelihood, different variants of the Wasserstein distance [14], Fréchet Inception Distance (FID) [15] and Maximum Mean Discrepancy (MMD), there is no consensus as to which measure best captures the strengths and limitations of generative models and should be used for a fair model comparison. Indeed, there are a number of desired properties for a good measure, including the ability to distinguish generated samples from real ones, favoring models that generate diverse samples, and having low computational and sample complexity. It is unlikely that a single measure can cover all aspects. Since different applications require different trade-offs among the desired properties, it has been argued that the evaluation metric should be chosen with respect to specific applications [16]. On the other hand, previous works have shown through empirical studies that MMD performs well in terms of the discriminability, robustness and efficiency compared to other metrics when it operates in the feature space [17,18]. Moreover, the empirical estimate of MMD enjoys favorable statistical properties such as asymptotic normality, making it a favorable choice to construct two-sample and three-sample tests that compares probability distributions. Therefore, we base our model selection criterion on MMD.

MMD is a metric of probability measures which falls within the family of integral probability metrics (IPMs) [19]. For two probability measures, $\mathbb{P}$ and $\mathbb{Q}$, over $\mathcal{X} \subset \mathbb{R}^d$, IPM is defined as

$$\mathcal{D}_{\mathcal{F}}(\mathbb{P}, \mathbb{Q}) = \sup_{f \in \mathcal{F}} \mathbb{E}_{\mathbb{P}} f(X) - \mathbb{E}_{\mathbb{Q}} f(Y), \tag{1}$$

which is the maximum difference between the mean function values on the two probability measures. The choice of the witness function class $\mathcal{F}$ determines the probability metric. The MMD is defined as the IPM with $\mathcal{F}$ being the unit ball in a reproducing kernel Hilbert space (RKHS) $\mathcal{H}$, with a positive definite kernel $k(\cdot, \cdot) : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$,

$$\mathrm{MMD}(\mathbb{P}, \mathbb{Q}; \mathcal{H}) = \sup_{f \in \mathcal{H}, \|f\|_{\mathcal{H}} \leq 1} \mathbb{E}_{\mathbb{P}} f(X) - \mathbb{E}_{\mathbb{Q}} f(Y).$$

It can be interpreted as the distance between the mean embeddings of $\mathbb{P}$ and $\mathbb{Q}$ into $\mathcal{H}$. It can be shown that the square of the MMD can be expressed as

$$\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}; \mathcal{H}) = \mathbb{E}_{\mathbb{P} \otimes \mathbb{P}}\left[k\left(X, X'\right)\right] - 2\mathbb{E}_{\mathbb{P} \otimes \mathbb{Q}}[k(X, Y)] + \mathbb{E}_{\mathbb{Q} \otimes \mathbb{Q}}\left[k\left(Y, Y'\right)\right], \tag{2}$$

where X and X' are independent random variables having distribution $\mathbb{P}$, and Y and Y' are independent random variables having distribution $\mathbb{Q}$ [13]. It immediately follows that MMD has a straightforward unbiased empirical estimator:

$$\begin{aligned} \mathrm{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_n, \mathcal{H}) = & \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j \neq i}^{m} k(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} k(y_i, y_j) \\ & - \frac{2}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} k(x_i, y_j), \end{aligned} \tag{3}$$

where $\mathbf{X}_m := \{x_1, \ldots, x_m\}$ and $\mathbf{Y}_n := \{y_1, \ldots, y_n\}$ are i.i.d. samples from $\mathbb{P}$ and $\mathbb{Q}$, respectively. Let $v_i := (x_i, y_i)$, $i = 1, \ldots, m$ be i.i.d samples from $\mathbb{P} \times \mathbb{Q}$, when $m = n$. Then,

$$\mathrm{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_m, \mathcal{H}) = \frac{1}{(m)(m-1)} \sum_{i \neq j}^{m} h(v_i, v_j) \tag{4}$$

is a U-statistic with $h(v_i, v_j) = k(x_i, x_j) + k(y_i, y_j) - k(x_i, y_j) - k(x_j, y_i)$. According to the properties of U-statistics, $\sqrt{m}\left(\mathrm{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_m, \mathcal{H}) - \mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}, \mathcal{H})\right)$ converges weakly to a Gaussian distribution as $m \to \infty$, when $\mathbb{P} \neq \mathbb{Q}$ and $\mathbf{E}(h^2) < \infty$. MMD and the Wasserstein distance are two extremes of the Sinkhorn divergences, an entropic regularized variant of the Wasserstein distance, e.g., [20,21].

Instead of simply using the MMD^2 at the current iteration to represent the intermediate performance of each candidate model, we consider an exponentially weighted average of MMD^2 that takes into account the model performance at previous training iterations, i.e.,

$$\tilde{\ell}_{k,R}^{\beta,h} = \frac{\sum_{r=0}^{h-1} \beta^r \, \mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}_k^{R-r}, \mathcal{H})}{\sum_{r=0}^{h-1} \beta^r}, \quad \text{for } k = 1, \ldots, K \text{ and } R > h-1, \tag{5}$$

where $h \geq 1$ is the window size, $\beta \in (0,1)$ is the decay rate, $\mathbb{P}$ is the reference/target distribution and $\mathbb{Q}_k^R$ denotes the distributions of the samples generated from model k after being trained for R units of resources. An exponentially weighted average smooths the learning curves. As a result, the models' performance represented by the smoothed MMD^2 are more distinguishable from each other. Figure 1 shows an illustrative example. Two Generative Adversarial Network (GAN) models with different training objectives are trained on the Half Moons dataset, respectively (details of the dataset can be found in Section 3). At the early stage of training, the variation of the loss tends to be large. While any sequences of losses (under the convergence assumption) would eventually stabilize and be separated from each other even without smoothing, smoothed loss is able to distinguish between the two models at a much earlier stage, which can help us make the right decision about which models should be trained further.

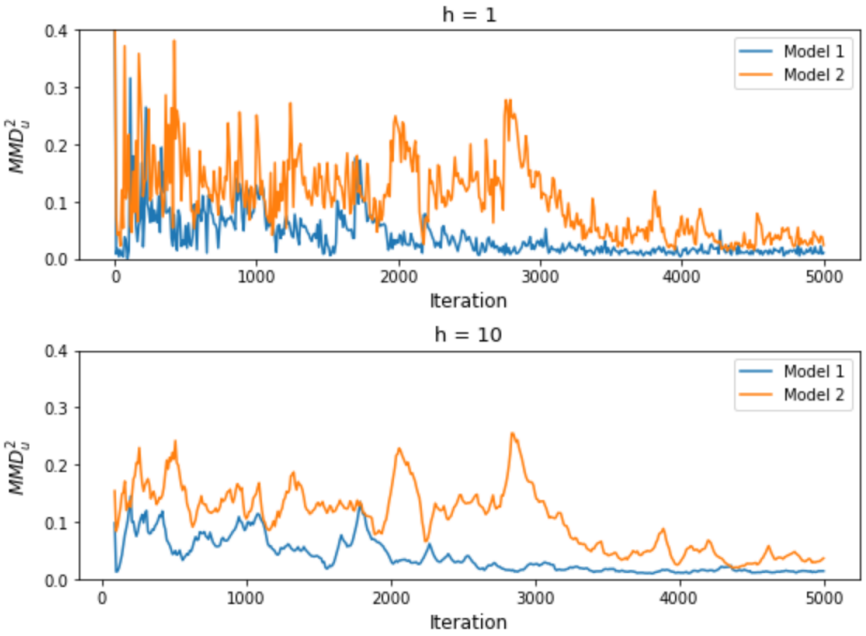

Figure 1. Upper: MMD_u^2 versus training iteration; **lower**: exponentially weighted average of MMD_u^2 with $h = 10$ and $\beta = 0.9$ versus training iteration.

2.3. Adaptive Successive Halving with Hypothesis Testing

Now, we introduce our proposed algorithm called Adaptive Successive Halving (AdaptSH), shown in Algorithm 2. As the name of our algorithm suggests, instead of following a predetermined elimination schedule as in Successive Halving, we base our decision on test results and adaptively change the elimination schedule based on the remaining budget and remaining number of arms. In particular, the major difference between AdaptSH and the original Successive Halving lies in line 8 of Algorithm 2, where

we perform a sequence of statistical tests to compare the intermediate performance of the current "best" arm and each of other remaining arms by taking samples from them and compare their relative similarity to the target distribution. The corresponding p-values are adjusted using the procedure proposed by Benjamini and Yekutieli [22], which controls the false discovery rate. To avoid inflating the type I error rate, we use two independent samples from each model to sort the models (line 7) and to perform the statistical tests (line 8), respectively. The algorithm stops allocating further resources to models that perform significantly worse than the current "best" model, as measured by a desired loss criteria (e.g., MMD). It should be noted that our proposed algorithm can perhaps be used if an alternative loss (e.g., the Wasserstein distance and alike) is chosen, but developing an asymptotic theory required by the statistical tests can be challenging.

Algorithm 2 Adaptive Successive Halving

Input: Budget B, K models $\mathbf{M}_1, \ldots, \mathbf{M}_K$, decay rate β, window size h, significance level α

1: $S_0 = \{1, 2, \ldots, K\}$
2: Initialize $i = 0$
3: $n = K$
4: **while** $B > 0$ **and** $n >= 2$ **do**
5: Allocate $r_i = \left\lfloor \frac{B}{n\lceil \log_2(n) \rceil} \right\rfloor$ units of resource to train each model in S_i
6: $R_i = \sum_{j=0}^{i} r_j$
7: Sort the intermediate losses of the models in S_i such that $\tilde{\ell}^{\beta,h}_{\sigma_i(1),R_i} \leq \tilde{\ell}^{\beta,h}_{\sigma_i(2),R_i} \leq \ldots \leq \tilde{\ell}^{\beta,h}_{\sigma_i(n),R_i}$, where $\sigma_i(\cdot)$ is a bijection from {1, 2, . . . , n} to S_i
8: Compare $M_{\sigma_i(1)}$ against $M_{\sigma_i(j)} (j = 2, \ldots, n)$ using three-sample tests comparing the relative closeness of their generated samples to the validation dataset with Benjamini and Yekutieli [22] correction to obtain the adjusted p-values $p_2^{\text{adjusted}}, \ldots, p_n^{\text{adjusted}}$
9: $S_{i+1} = \{\sigma_i(j) \mid 2 \leq j \leq n \text{ and } p_j^{\text{adjusted}} > \alpha\} \cup \{\sigma_i(1)\}$
10: $B = B - nr_i$
11: $n = |S_{i+1}|$
12: $i = i + 1$
13: **end while**

The statistical test in line 8 is a three-sample relative similarity test that aims to determine with high significance whether the samples generated by the current "best" model are closer to the evaluation data set than those of each remaining model. While there is rich literature on two-sample test problems for multivariate data, statistical tests for three-sample relative similarity are rarely studied in the literature. Bounliphone et al. [23] propose a relative similarity test:

$$\mathcal{H}_0 : \text{MMD}^2(\mathbb{P}, \mathbb{Q}, \mathcal{H}) = \text{MMD}^2(\mathbb{P}, \mathbb{T}, \mathcal{H})$$
$$\mathcal{H}_1 : \text{MMD}^2(\mathbb{P}, \mathbb{Q}, \mathcal{H}) < \text{MMD}^2(\mathbb{P}, \mathbb{T}, \mathcal{H}),$$

which tests the null hypothesis that two distributions $\mathbb{Q}$ and $\mathbb{T}$ are equally close to a target distribution $\mathbb{P}$ against the alternative hypothesis that $\mathbb{Q}$ is closer to $\mathbb{P}$ than $\mathbb{T}$. They propose the following test statistic:

$$\text{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_m, \mathcal{H}) - \text{MMD}_u^2(\mathbf{X}_m, \mathbf{Z}_m, \mathcal{H}), \tag{6}$$

where $\mathbf{X}_m := \{x_1, \ldots, x_m\}$, $\mathbf{Y}_m := \{y_1, \ldots, y_m\}$ and $\mathbf{Z}_m := \{z_1, \ldots, z_m\}$ are iid samples from $\mathbb{P}$, $\mathbb{Q}$ and $\mathbb{T}$, respectively. The test statistic is asymptotically Gaussian, which directly follows from Hoeffding [24] (Theorem 7.1), which states that the joint distribution of several U-statistics converges weakly to a multivariate Gaussian distribution as $m \to \infty$.

We generalize their test to compare the averages of MMD^2 between two arms. In Section 2.2, we defined the exponentially weighted MMD^2, $\tilde{\ell}^{\beta,h}_{k,R}$ that we use to represent

the performance of arm k after being trained for R units of resources. In particular, we want to determine with high significance whether one arm has a smaller exponentially weighted MMD^2 than the other, which translates to the following null and alternative hypothesis:

$$\mathcal{H}_0 : \sum_{r=0}^{h-1} \beta^r \,\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}_1^{R-r}, \mathcal{H}) = \sum_{r=0}^{h-1} \beta^r \,\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}_2^{R-r}, \mathcal{H})$$
$$\mathcal{H}_1 : \sum_{r=0}^{h-1} \beta^r \,\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}_1^{R-r}, \mathcal{H}) < \sum_{r=0}^{h-1} \beta^r \,\mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}_2^{R-r}, \mathcal{H}),$$

where $\{\mathbb{Q}_i^R\}_{R>0}, i = 1, 2$ denote two sequences of distributions corresponding to two arms that are being compared. It is then natural to use the following test statistic:

$$T_m^{h,\beta} = \sum_{r=0}^{h-1} \beta^r \,\mathrm{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_m^{1,R-r}, \mathcal{H}) - \sum_{r=0}^{h-1} \beta^r \,\mathrm{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_m^{2,R-r}, \mathcal{H}), \tag{7}$$

where $\mathbf{X}_m := \{x_1, \ldots, x_m\}$, $\mathbf{Y}_m^{1,R} := \{y_1^{1,R}, \ldots, y_m^{1,R}\}$ and $\mathbf{Y}_m^{2,R} := \{y_1^{2,R}, \ldots, y_m^{2,R}\}$ are iid samples from $\mathbb{P}$, $\mathbb{Q}_1^R$ and $\mathbb{Q}_2^R$ respectively. The following theorem states the asymptotic normality of the joint distribution of multiple unbiased estimators of MMD^{2}s, which follows directly from Hoeffding [24] (Theorem 7.1).

Theorem 1. *Assume that* $\mathbf{E}_{v,v' \sim \mathbb{P} \times \mathbb{Q}_i^r} h^2(v, v') < \infty$ *and* $\mathbb{P} \neq \mathbb{Q}_i^r$ *for* $i = 1, 2$ *and* $r = R - h + 1, \ldots, R$, *then*

$$\sqrt{m}\left(\begin{pmatrix} \mathrm{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_m^{1,R}, \mathcal{H}) \\ \cdots \\ \mathrm{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_m^{1,R-h+1}, \mathcal{H}) \\ \mathrm{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_m^{2,R}, \mathcal{H}) \\ \cdots \\ \mathrm{MMD}_u^2(\mathbf{X}_m, \mathbf{Y}_m^{2,R-h+1}, \mathcal{H}) \end{pmatrix} - \begin{pmatrix} \mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}_1^R, \mathcal{H}) \\ \cdots \\ \mathrm{MMD}^2\left(\mathbb{P}, \mathbb{Q}_1^{R-h+1}, \mathcal{H}\right) \\ \mathrm{MMD}^2(\mathbb{P}, \mathbb{Q}_2^R, \mathcal{H}) \\ \cdots \\ \mathrm{MMD}^2\left(\mathbb{P}, \mathbb{Q}_2^{R-h+1}, \mathcal{H}\right) \end{pmatrix}\right) \xrightarrow{d} \mathcal{N}\left(\mathbf{0}_{2h}, \mathbf{\Sigma}_m^{h,\beta}\right), \tag{8}$$

where $\mathbf{0}_{2h}$ *denotes a vector of zeros with length* $2h$, *and* $\mathbf{\Sigma}_m^{h,\beta}$ *denotes the covariance matrix.*

The explicit form of $\mathbf{\Sigma}_m^{h,\beta}$ and its empirical estimate are given in Appendix A. Then, the p-value can be approximated by

$$p \approx \Phi\left(\frac{T_m^{h,\beta}}{\sqrt{\frac{1}{m}\boldsymbol{\beta}_h^T \mathbf{\Sigma}_m^{h,\beta} \boldsymbol{\beta}_h}}\right), \tag{9}$$

where $\boldsymbol{\beta}_h = (1, \beta^2, \ldots, \beta^{h-1}, -1, -\beta^2, \ldots, -\beta^{h-1})^T$ and $\Phi(\cdot)$ is the cumulative distribution function of a standard normal distribution. When $h = 1$, our proposed test reduces to the three-sample relatively similarity test proposed in Bounliphone et al. [23]. Notice that as we have a closed form expression to compute the p-values using the asymptotic distribution (as given in the Appendix A), there is not much additional overhead in using the statistical tests within our proposed AdaptSH compared to the traditional SH.

We use the same example as we used in Section 2.2 to illustrate the effect of using exponentially weighted average on the results of the statistical test between arms. We apply the test based only on the current MMD^2 and our proposed test respectively to the two models shown in Figure 1. The tests are performed every 10 iterations, and the alternative hypothesis considered is that Model 1 (represented in blue in Figure 1) has smaller (exponentially weighted) MMD^{2}s. The resulting p-values are shown in Figure 2. Suppose the significance level $\alpha = 0.01$ is considered. Then, a p-value less than 0.01 indicates that the Model 2 is significantly worse than Model 1 and will be stopped from further

training based on our model search algorithm. And a p-value greater than 0.99 indicates that Model 2 is significantly better then Model 1 and that Model 1 will be early stopped, since it is equivalent to a p-value less than 0.01 if the opposite alternative hypothesis were considered. Figure 3 shows that increasing h increases the ease of making the right decision during the training process. For this particular example, when $h \geq 7$, the chance of early stopping in favor of the worse model reduces to zero.

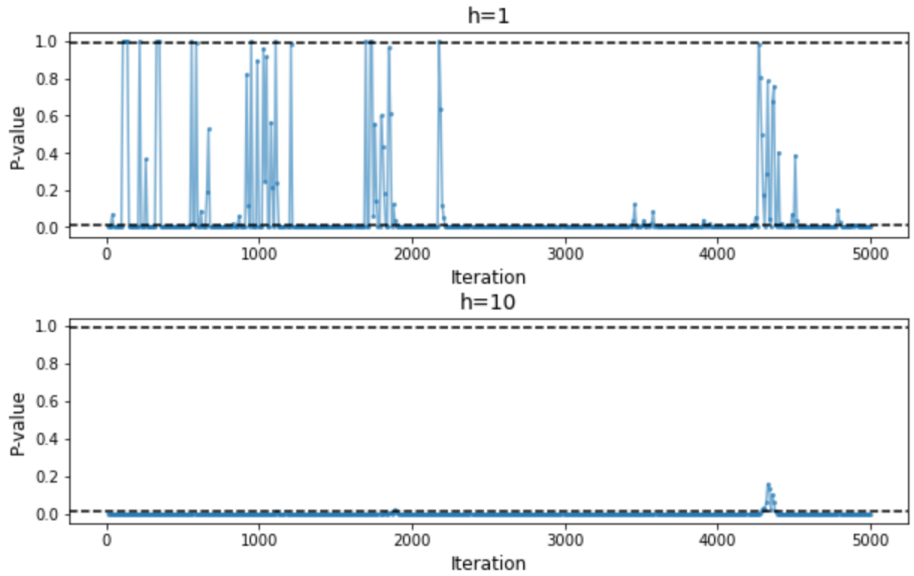

Figure 2. Upper: p-values of the statistical tests when only the most recent MMD^2 is considered; **lower**: p-values of the statistical tests when historical MMD^2s within the moving window with $h = 10$ and $\beta = 0.9$ are considered. The two horizontal dashed lines correspond to p-values of 0.01 and 0.99 respectively.

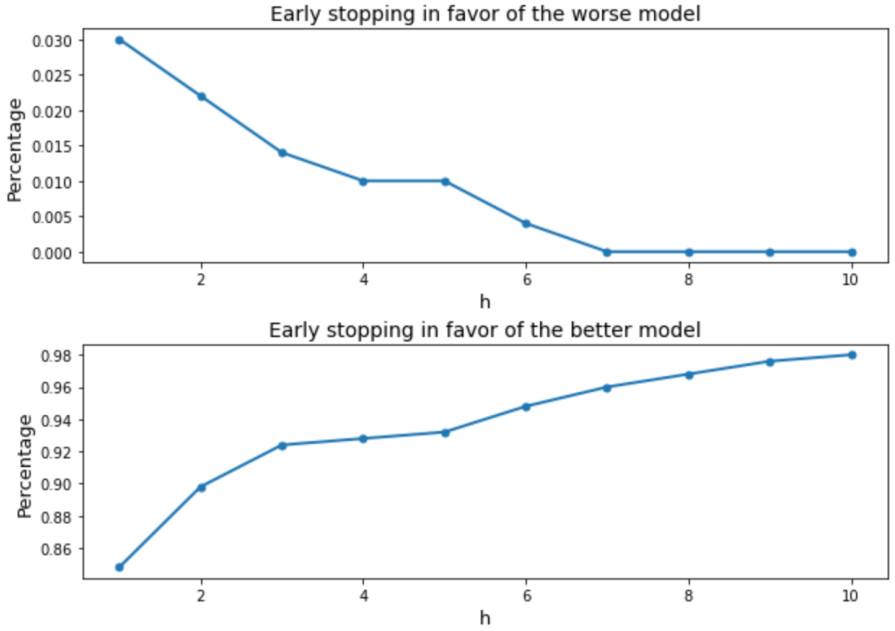

Figure 3. Upper: The change of the percentage of tests that have a p-value less than 0.01 with h; **lower**: The change of the percentage of tests that have a p-value greater than 0.99 with h.

3. Experimental Results

In this section, we compare our proposed algorithm to Successive Halving on two hyperparameter optimization problems for GAN models. In particular, we consider a number

of GAN models that are trained using different variants of the Sliced Wasserstein distance. The space of the models to search includes the set of distance metrics to search over and the reasonable ranges of their associated hyperparameters. We consider seven different variants of the Sliced Wasserstein distance and a set of different combinations of hyperparameters for each of them, which sum up to 30 GAN models in total. The details of the hyperparameter configurations considered for each distance metric can be found in Appendix B. The same generator and discriminator architectures are used for all 30 models (See Appendix B).

To evaluate the different search algorithms' performance under different budgets, we set the total budget of the iterations to a sequence of values, and for each budget let the search algorithms decide how to allocate it amongst the different arms. The final performance of the models is represented by their final losses, defined as $\nu_k = \lim_{j\to\infty} \ell_{k,j}$ for each model k. As ν_k is unknown, and we approximate it by the loss after training the model for some finite units of resources R. In real-world applications, R is often determined by the maximum amount of resources that one wants to allocate to any given configuration, which are often inferred from previous training experiences on similar tasks or determined by the time and money one wants to spend on training one model for a particular task. To address the oscillation around the optimal value when the models are trained by stochastic optimization methods, we use the average loss within a small window $[R, R+\Delta)$ to approximate $\nu_k, k = 1, \ldots, K$, i.e, $\hat{v}_k = \frac{1}{\Delta}\sum_{j=R}^{R+\Delta-1} \hat{\ell}_{k,j}$.

2-D Half Moons We first apply the searching algorithms to compare the 30 GAN models being trained on the Half Moons dataset. We uniformly sample 1000 points on the half moons as the training set and another 1000 points as the validation set. These sample sizes are used for just for numerical illustration but the results are not sensitive to these choices unless we use very small sample sizes. The training data are shown in Figure 4. At each elimination round, 500 points are sampled from each model to compute MMD_u^2 that is used for sorting, and another 500 points are sampled for statistical testing. We set one unit of resources to 10 training iterations. And we vary the budget from 400 to 1400 units by a step of 50. And for each budget, we record the estimated final loss of the selected arm by each searching algorithm. We repeat the experiment for 20 trials. For each searching algorithm and each budget, we compute the average loss of the 20 trials, $\hat{\mu}_{i,B} = \frac{1}{20}\sum_{j=1}^{20} \hat{v}^j_{k(i,B,j)}$, where i, B and j denote the search algorithm, budget and trial number, respectively, $k(i, B, j)$ denotes the index of the model selected by algorithm i using budget B at trial j, and $\hat{v}^j_k$ denotes the estimated final loss of the model k at trial j. The tuning parameters for the exponential weighting are chosen as $\beta = 0.9$ and $h = 6$. For the choice of the kernel function in MMD, we follow Bounliphone et al. [23] and use a Gaussian kernel with bandwidth selected as the median pairwise distance between data points. But one can use alternative kernel functions (e.g., the semantic-aware deep kernel proposed by Gao et al. [25]). Additional numerical results for other tuning parameter settings are provided in Appendix C.

As is shown in Figure 5, for the majority of the budgets considered, our proposed method selects arms with smaller final losses on average. To further compare the losses of the arms selected by the two searching algorithms, we conduct a Mann–Whitney U test between the two samples $\{\hat{v}^j_{k(1,B,j)}\}_{j=1}^{20}$ and $\{\hat{v}^j_{k(2,B,j)}\}_{j=1}^{20}$ for each B with the alternative hypothesis that the final loss obtained by AdaptSH is stochastically smaller than that of Successive Halving. The p-values of the tests are shown in Figure 5. For budgets over 600, the p-value stays around 0.05, which indicates that our algorithm outperforms SH with a high confidence for a large range of budgets.

3-D Swiss Roll We next compare AdaptSH to Successive Halving on the Swiss Roll dataset. A total of 1000 points are sampled on the Swiss Roll as the training set and another 2000 points as the validation set. The training data are shown in Figure 6. At each elimination round, 1000 points are sampled from each model to compute MMD_u^2 that is used for sorting, and another 1000 points are sampled for statistical testing. The experiment

setting is the same as that in the previous example, as well as the choice of the tuning parameters for the exponential weighting. We repeat the experiment for 20 trials. And the results are shown in Figure 7. Our proposed method selects arms with smaller final losses on most of the budgets, and the advantage is significant for most moderate budgets.

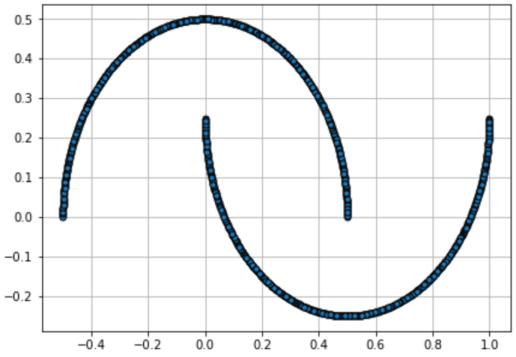

Figure 4. One thousand points sampled on the "Half Moons".

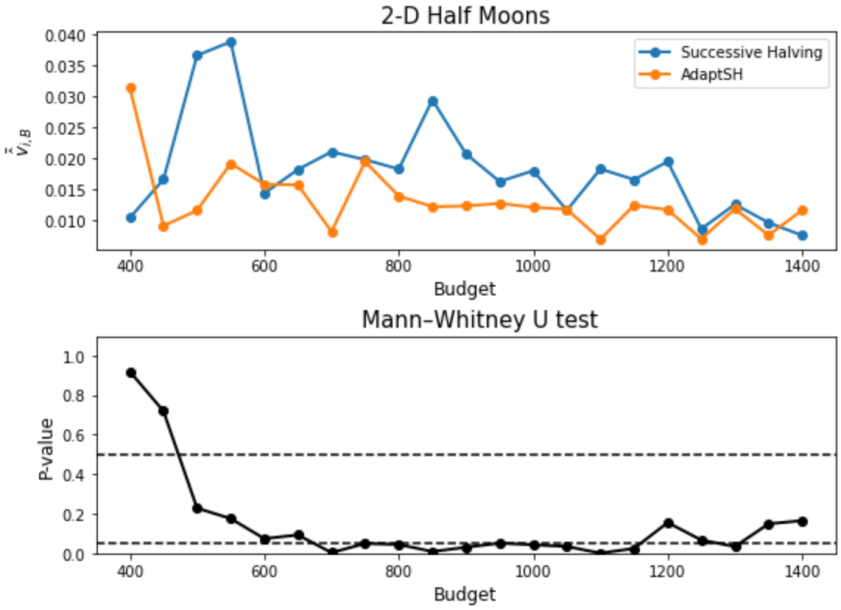

Figure 5. Results on the 2-D Half Moons dataset. **Upper**: Final loss of the selected models averaged over 20 trials against the input budget. **Lower**: p-values of the Mann–Whitney U tests. Two horizontal dashed lines represent p-value = 0.05 and 0.5, respectively.

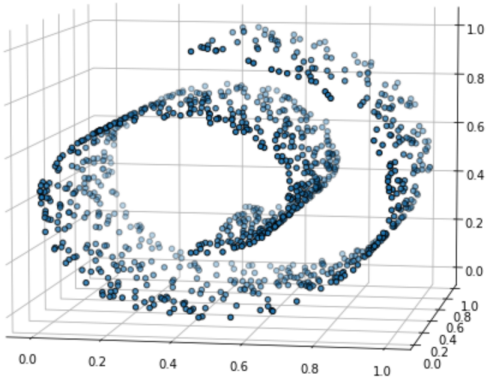

Figure 6. One thousand points sampled on the "Swiss Roll".

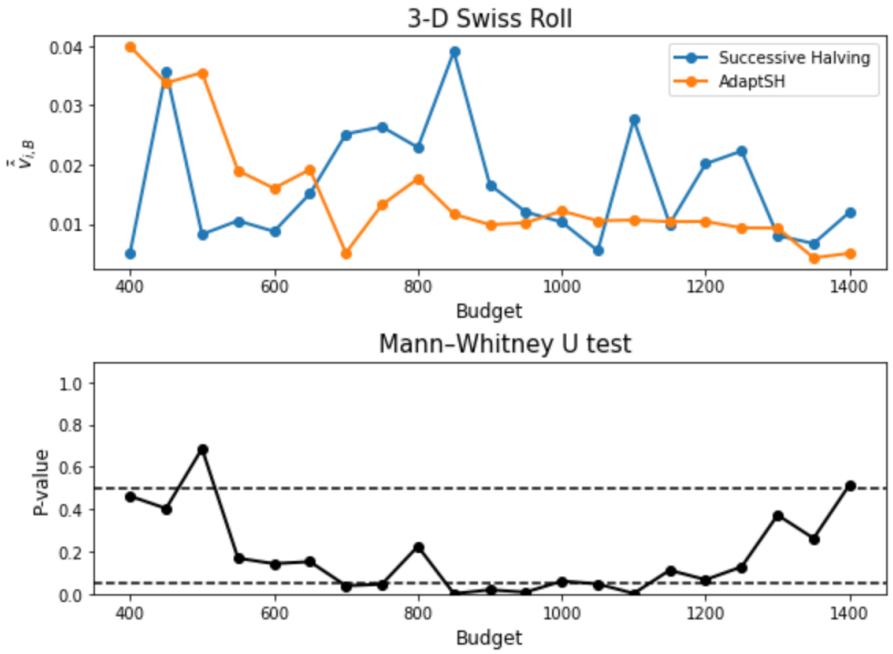

Figure 7. Results on the 3-D Swiss Roll dataset. **Upper**: Final loss of the selected model averaged over 20 trials against the input budget. **Lower**: p-values of the Mann–Whitney U tests. Two horizontal dashed lines represent p-value = 0.05 and 0.5 respectively.

4. Conclusions

We have presented a method for hyperparameter optimization for generative models. We cast it as a non-stochastic best-arm identification problem with a fixed budget and identify the clearly underperforming arms early on through statistical tests between average empirical MMD^2s of partially trained models. In our experiments, we showed that our procedure leads to a significant improvement in the performance of the selected configurations compared to Successive Halving on a wide range of budgets and is robust to the choice of the tuning parameters. Our method does not make any assumptions about the parametric form of the learning curves, nor about their monotonicity or smoothness, which makes it general enough to be suitable for most hyperparameter optimization tasks.

Author Contributions: Conceptualization, L.C. and S.K.G.; methodology, L.C. and S.K.G.; software, L.C.; validation, L.C.; formal analysis, L.C.; investigation, L.C.; resources, L.C. and S.K.G.; data curation, L.C.; writing—original draft preparation, L.C.; writing—review and editing, L.C. and S.K.G.; visualization, L.C.; supervision, S.K.G.; project administration, S.K.G.; funding acquisition, not applicable. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Datasets used in this paper were generated during the study. The code to generate the datasets is available upon request.

Conflicts of Interest: The authors declare no conflicts of interest.

Appendix A. Derivation of the Covariance Matrix

The variance and covariance of MMD_u^2 are derived and in Bounliphone et al. ([23], Appendix A), as well as their sample estimates. Here, we extend their results and give the explicit expression for our covariance matrix $\mathbf{\Sigma}_m^{h,\beta}$ in Equation (8).

Let $\phi(x) = k(x, \cdot) : \mathcal{X} \to \mathbb{R}$, a feature mapping from $\mathcal{X}$ to $\mathbb{R}$, and $\mu_{\mathbb{P}}(\cdot) = \mathbf{E}_{x\sim\mathbb{P}} k(x, \cdot) \in \mathcal{H}$, the mean embedding of the distribution $\mathbb{P}$ into $\mathcal{H}$.

Then the diagonal terms of $\mathbf{\Sigma}_m^{h,\beta}$ are given by

$$\sigma_{ii} = \begin{cases} 4\zeta^{\mathbb{P},\mathbb{Q}_1^{R-i+1}} & \text{if } 1 \le i \le h \\ 4\zeta^{\mathbb{P},\mathbb{Q}_2^{R-i+1+h}} & \text{if } h < i \le 2h \end{cases},$$

where

$$\begin{aligned} \zeta^{\mathbb{P},\mathbb{Q}} =& \mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{\mathbb{P}}\rangle_{\mathcal{H}}^2\right] - \mathrm{E}_{x\sim\mathbb{P}}[\langle\phi(x),\mu_{\mathbb{P}}\rangle_{\mathcal{H}}]^2 \\ &- 2\left(\mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{\mathbb{P}}\rangle_{\mathcal{H}}\langle\phi(x),\mu_{\mathbb{Q}}\rangle_{\mathcal{H}}\right] - \mathrm{E}_{x\sim\mathbb{P}}[\langle\phi(x),\mu_{\mathbb{P}}\rangle_{\mathcal{H}}]\mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{\mathbb{Q}}\rangle_{\mathcal{H}}\right]\right) \\ &+ \mathrm{E}_{y\sim\mathbb{Q}}\left[\langle\phi(y),\mu_{\mathbb{Q}}\rangle_{\mathcal{H}}^2\right] - \mathrm{E}_{y\sim\mathbb{Q}}\left[\langle\phi(y),\mu_{\mathbb{Q}}\rangle_{\mathcal{H}}\right]^2 \\ &- 2\left(\mathrm{E}_{y\sim\mathbb{Q}}\left[\langle\phi(y),\mu_{\mathbb{Q}}\rangle_{\mathcal{H}}\langle\phi(y),\mu_{\mathbb{P}}\rangle_{\mathcal{H}}\right] - \mathrm{E}_{y\sim\mathbb{Q}}\left[\langle\phi(y),\mu_{\mathbb{Q}}\rangle_{\mathcal{H}}\right]\mathbb{E}_{y\sim\mathbb{Q}}[\langle\phi(y),\mu_{\mathbb{P}}\rangle_{\mathcal{H}}]\right) \\ &+ \mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{\mathbb{Q}}\rangle_{\mathcal{H}}^2\right] - \mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{y}\rangle_{\mathcal{H}}\right]^2 \\ &+ \mathrm{E}_{y\sim\mathbb{Q}}\left[\langle\phi(y),\mu_{\mathbb{P}}\rangle_{\mathcal{H}}^2\right] - \mathrm{E}_{y\sim\mathbb{Q}}[\langle\phi(y),\mu_{\mathbb{P}}\rangle_{\mathcal{H}}]^2 \end{aligned} \tag{A1}$$

with sample estimate

$$\begin{aligned} \hat{\zeta}^{\mathbb{P},\mathbb{Q}} =& \frac{1}{m(m-1)^2}e^T\tilde{K}_{xx}\tilde{K}_{xx}e - \left(\frac{1}{m(m-1)}e^T\tilde{K}_{xx}e\right)^2 \\ &- 2\left(\frac{1}{m^2(m-1)}e^T\tilde{K}_{xx}K_{xy}e - \frac{1}{m^3(m-1)}e^T\tilde{K}_{xx}ee^TK_{xy}e\right) \\ &+ \frac{1}{m(m-1)^2}e^T\tilde{K}_{yy}\tilde{K}_{yy}e - \left(\frac{1}{m(m-1)}e^T\tilde{K}_{yy}e\right)^2 \\ &- 2\left(\frac{1}{m^2(m-1)}e^T\tilde{K}_{yy}K_{yx}e - \frac{1}{n^2(m-1)m}e^T\tilde{K}_{yy}ee^TK_{xy}e\right) \\ &+ \frac{1}{m^3}e^TK_{yx}K_{xy}e - 2\left(\frac{1}{m^2}e^TK_{xy}e\right)^2 + \frac{1}{m^3}e^TK_{xy}K_{yx}e, \end{aligned} \tag{A2}$$

where e is a vector of 1s with appropriate size, K_{xx}, K_{yy} and K_{xy} are the kernel matrices, and $\tilde{K}$ indicates that the diagonal terms have been set to zero. For example, $\left[\tilde{K}_{xx}\right]_{ij} = [K_{xx}]_{ij}$ for all $i \neq j$ and $\left[\tilde{K}_{xx}\right]_{ij} = 0$ for $i = j$.

And the off-diagonal terms of $\mathbf{\Sigma}_m^{h,\beta}$ are given by

$$\sigma_{ij} = \begin{cases} 4\zeta^{\mathbb{P},\mathbb{Q}_1^{R-i+1},\mathbb{Q}_1^{R-j+1}} & \text{if } 1 \le i,j \le h \\ 4\zeta^{\mathbb{P},\mathbb{Q}_1^{R-i+1},\mathbb{Q}_2^{R-j+1+h}} & \text{if } 1 \le i \le h \text{ and } h < j \le 2h \\ 4\zeta^{\mathbb{P},\mathbb{Q}_2^{R-i+1+h},\mathbb{Q}_2^{R-j+1+h}} & \text{if } h < i,j \le 2h \end{cases},$$

where

$$\begin{aligned} \zeta^{\mathbb{P},\mathbb{Q},\mathbb{T}} =& \mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{\mathbb{P}}\rangle^2\right] - \mathrm{E}_{x\sim\mathbb{P}}[\langle\phi(x),\mu_{\mathbb{P}}\rangle]^2 \\ &- \left(\mathrm{E}_{x\sim\mathbb{P}}[\langle\phi(x),\mu_{\mathbb{P}}\rangle\langle\phi(x),\mu_{\mathbb{T}}\rangle] - \mathrm{E}_{x\sim\mathbb{P}}[\langle\phi(x),\mu_{\mathbb{P}}\rangle]\mathrm{E}_{x\sim\mathbb{P}}[\langle\phi(x),\mu_{\mathbb{T}}\rangle]\right) \\ &- \left(\mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{\mathbb{P}}\rangle\langle\phi(x),\mu_{\mathbb{Q}}\rangle\right] - \mathrm{E}_{x\sim\mathbb{P}}[\langle\phi(x),\mu_{\mathbb{P}}\rangle]\mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{\mathbb{Q}}\rangle\right]\right) \\ &+ \mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{\mathbb{Q}}\rangle\langle\phi(x),\mu_{\mathbb{T}}\rangle\right] - \mathrm{E}_{x\sim\mathbb{P}}\left[\langle\phi(x),\mu_{\mathbb{Q}}\rangle\right]\mathrm{E}_{x\sim\mathbb{P}}[\langle\phi(x),\mu_{\mathbb{T}}\rangle], \end{aligned} \tag{A3}$$

with sample estimator

$$\begin{aligned}\hat{\zeta}^{\mathbb{P},\mathbb{Q},\mathbb{T}} =& \frac{1}{m(m-1)^2} e^T \tilde{K}_{xx} \tilde{K}_{xx} e - \left(\frac{1}{m(m-1)} e^T \tilde{K}_{xx} e\right)^2 \\ &- \left(\frac{1}{m^2(m-1)} e^T \tilde{K}_{xx} K_{xz} e - \frac{1}{m^3(m-1)} e^T \tilde{K}_{xx} e e^T K_{xz} e\right) \\ &- \left(\frac{1}{m^2(m-1)} e^T \tilde{K}_{xx} K_{xy} e - \frac{1}{m^3(m-1)} e^T \tilde{K}_{xx} e e^T K_{xz} e\right) \\ &+ \left(\frac{1}{m^3} e^T K_{yx} K_{xz} e - \frac{1}{m^4} e^T K_{xy} e e^T K_{xz} e\right).\end{aligned} \tag{A4}$$

Appendix B. Experiment Settings

Table A1 lists the distance metrics considered in the experiment as well as the choices of hyperparameters of each of them. Here, *niter* refers to the number of iterations used to compute the distance metric if it requires an optimization procedure. L denotes the number of the projections (slices) used to estimate the sliced distances. The definition of the remaining hyperparameters can be found in their corresponding original papers.

The generator architecture used for training the GAN models:

$$z \in \mathbb{R}^2 \rightarrow FC_{512} \rightarrow ReLU \rightarrow FC_{512} \rightarrow ReLU \rightarrow FC_{512} \rightarrow ReLU \rightarrow FC_{2(3)}$$

The discriminator architecture used for training the GAN models:

$$x \in \mathbb{R}^{2(3)} \rightarrow FC_{512} \rightarrow ReLU \rightarrow FC_{512} \rightarrow ReLU \rightarrow FC_{128} \rightarrow ReLU \rightarrow FC_1 \rightarrow Sigmoid$$

Table A1. Variants of the Sliced Wasserstein distance and their hyperparameter settings that are considered in the experiment.

Distance Metric	Hyperparameters
ASWD [26]	L: {1000}, *niter*: {10, 20}, λ: {1, 5}, *learningrate*: {0.005, 0.0005}
MSWD [27]	*niter*: {10, 50}
SWD [28]	L: {10, 1000}
GSWD [29]	L: {10, 1000}, r: {2, 5}
MGSWD [29]	*niter*: {10, 50}, r: {2, 5}
MGSWNN [29]	*niter*: {10, 50}
DSWD [30]	L: {1000}, *niter*: {10, 20}, λ_C: {1, 5}, *learning rate*: {0.005, 0.0005}

Appendix C. Additional Experimental Results

In this appendix, we provide additional experimental results to illustrate the robustness of our algorithm to the choice of tuning parameters h and β, and to yield a better understanding about the resource allocation behavior of our proposed algorithm. Figures A1–A4 show the results of using different combinations of the tuning parameters h and β on the Half Moons and the Swiss Roll dataset respectively. The results show that for all the combinations of the tuning hyperparameters, our algorithm outperforms Successive Halving on average for most of the budgets.

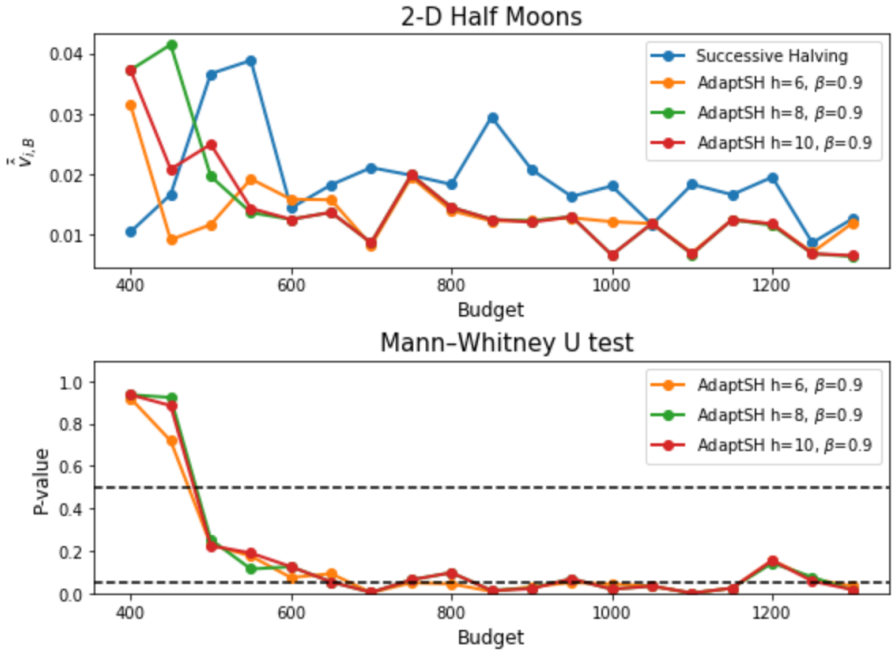

Figure A1. Results on the 2-D Half Moons dataset with $\beta = 0.9$ and varying h. **Upper**: Final loss of the selected models averaged over 20 trials against the input budget. **Lower**: p-values of the Mann–Whitney U tests. Two horizontal dashed lines represent p-value = 0.05 and 0.5 respectively.

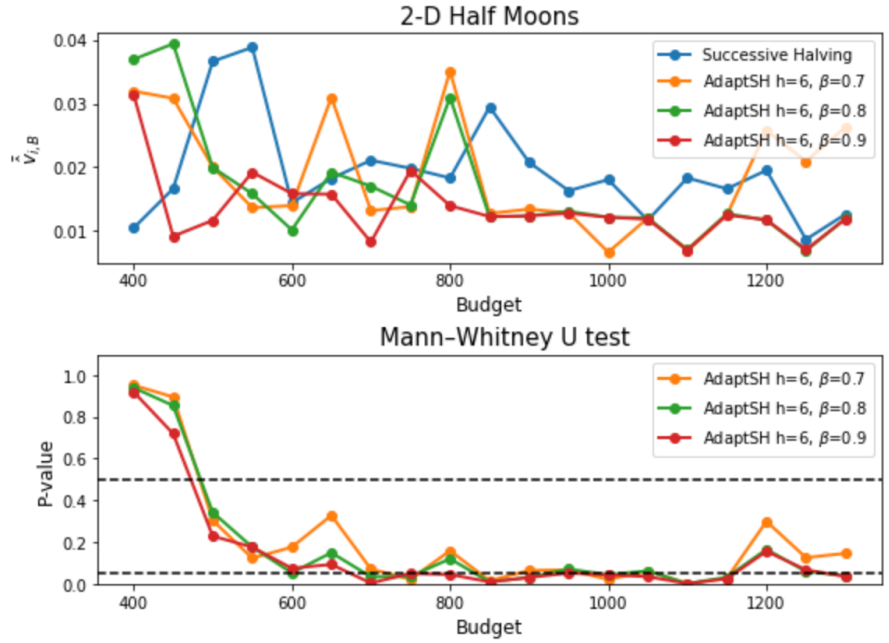

Figure A2. Results on the 2-D Half Moons dataset with $h = 6$ and varying β. **Upper**: Final loss of the selected models averaged over 20 trials against the input budget. **Lower**: p-values of the Mann–Whitney U tests. Two horizontal dashed lines represent p-value = 0.05 and 0.5 respectively.

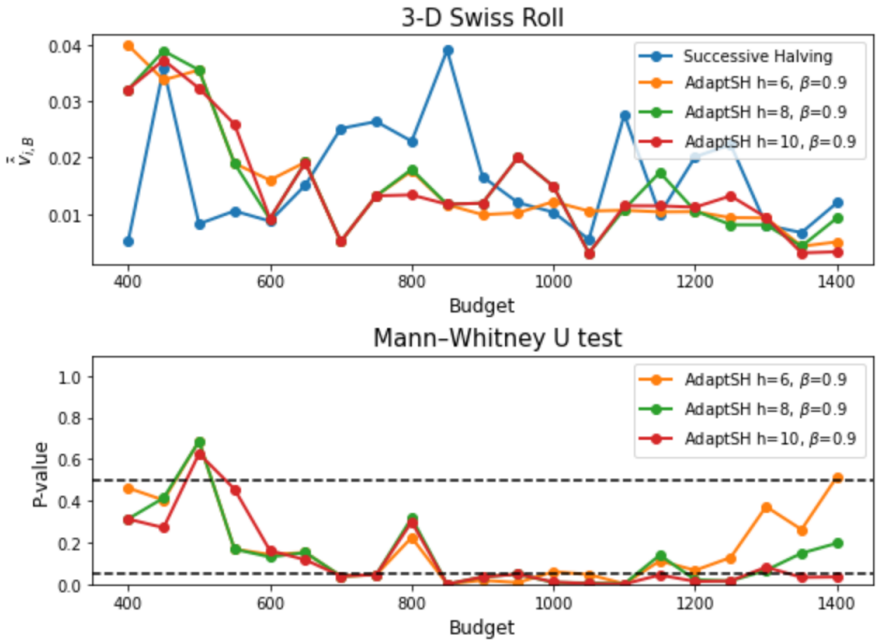

Figure A3. Results on the 3-D Swiss Roll dataset with $\beta = 0.9$ and varying h. **Upper**: Final loss of the selected models averaged over 20 trials against the input budget. **Lower**: p-values of the Mann–Whitney U tests. Two horizontal dashed lines represent p-value = 0.05 and 0.5 respectively.

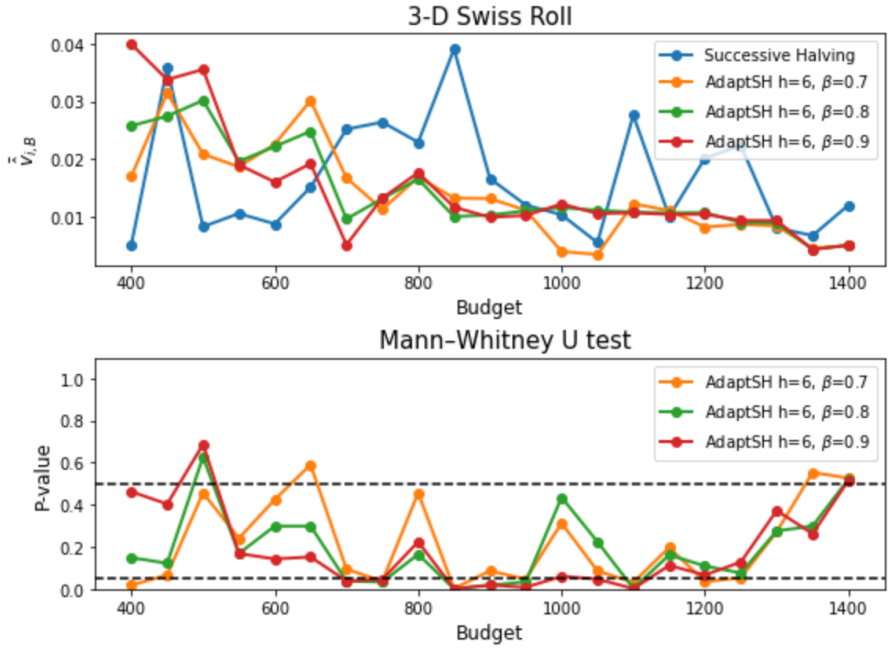

Figure A4. Results on the 3-D Swiss Roll dataset with $h = 6$ and varying β. **Upper**: Final loss of the selected models averaged over 20 trials against the input budget. **Lower**: p-values of the Mann–Whitney U tests. Two horizontal dashed lines represent p-value = 0.05 and 0.5 respectively.

Figure A5 illustrates the difference between the resource allocation of Successive Halving and that of AdaptSH. Successive Halving has exhausted the budget and has to make the final decision at around iteration 1000, while AdaptSH is able to train some of the models to 2000 iterations before making the final decision, which results in overall smaller MMDs of the selected models (see Figure A6).

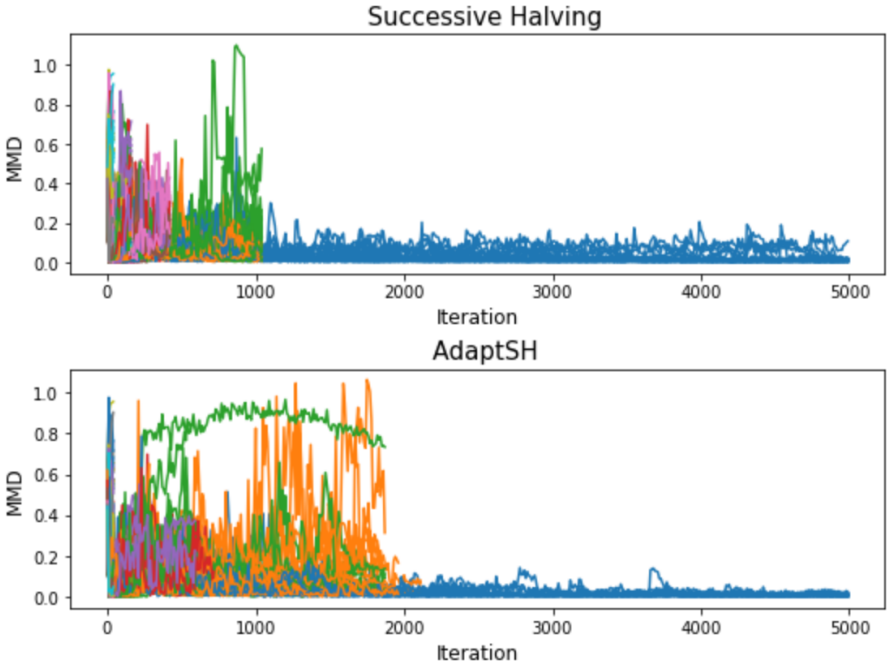

Figure A5. Comparison of learning curves of AdaptSH and Successive Halving on the Half Moons dataset with budget $B = 700$. The plots contain all learning curves from the 20 runs. Each curve is truncated at the time of early stopping. The blue curves represent the final selected models of the 20 runs respectively.

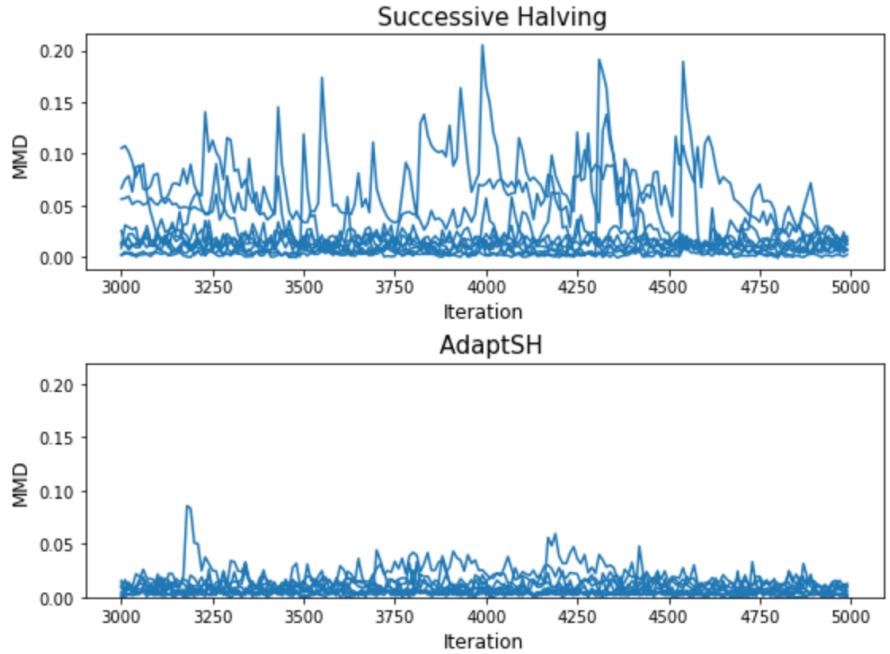

Figure A6. Iteration 3000 to 5000 of Figure A5.

References

1. Bergstra, J.; Bardenet, R.; Bengio, Y.; Kégl, B. Algorithms for hyper-parameter optimization. In Proceedings of the 25th Annual Conference on Neural Information Processing Systems (NIPS 2011), Granada, Spain, 12–15 December 2011; Neural Information Processing Systems Foundation: La Jolla, CA, USA, 2011; Volume 24.
2. Hutter, F.; Hoos, H.H.; Leyton-Brown, K. Sequential model-based optimization for general algorithm configuration. In Proceedings of the International Conference on Learning and Intelligent Optimization, Rome, Italy, 17–21 January 2011; Springer: Berlin/Heidelberg, Germany, 2011; pp. 507–523.

3. Snoek, J.; Larochelle, H.; Adams, R.P. Practical bayesian optimization of machine learning algorithms. *Adv. Neural Inf. Process. Syst.* **2012**, *25*. [CrossRef]
4. Wang, Z.; Zoghi, M.; Hutter, F.; Matheson, D.; De Freitas, N. Bayesian Optimization in High Dimensions via Random Embeddings. In Proceedings of the IJCAI, Beijing, China, 3–9 August 2013; pp. 1778–1784.
5. Swersky, K.; Snoek, J.; Adams, R.P. Freeze-thaw bayesian optimization. *arXiv* **2014**, arXiv:1406.3896.
6. Domhan, T.; Springenberg, J.T.; Hutter, F. Speeding up automatic hyperparameter optimization of deep neural networks by extrapolation of learning curves. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, Buenos Aires, Argentina, 25–31 July 2015.
7. Klein, A.; Falkner, S.; Bartels, S.; Hennig, P.; Hutter, F. Fast bayesian optimization of machine learning hyperparameters on large datasets. In Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Lauderdale, FL, USA, 20–22 April 2017; pp. 528–536.
8. Sparks, E.R.; Talwalkar, A.; Haas, D.; Franklin, M.J.; Jordan, M.I.; Kraska, T. Automating model search for large scale machine learning. In Proceedings of the Sixth ACM Symposium on Cloud Computing, Kohala Coast, HI, USA, 27–29 August 2015; pp. 368–380.
9. Jamieson, K.; Talwalkar, A. Non-stochastic best arm identification and hyperparameter optimization. In Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, Cadiz, Spain, 9–11 May 2016; pp. 240–248.
10. Li, L.; Jamieson, K.; DeSalvo, G.; Rostamizadeh, A.; Talwalkar, A. Hyperband: A novel bandit-based approach to hyperparameter optimization. *J. Mach. Learn. Res.* **2017**, *18*, 6765–6816.
11. Karnin, Z.; Koren, T.; Somekh, O. Almost optimal exploration in multi-armed bandits. In Proceedings of the International Conference on Machine Learning, Atlanta, GA, USA, 17–19 June 2013; pp. 1238–1246.
12. Wang, Z.; She, Q.; Ward, T.E. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Comput. Surv. (CSUR)* **2021**, *54*, 1–38. [CrossRef]
13. Gretton, A.; Borgwardt, K.M.; Rasch, M.J.; Schölkopf, B.; Smola, A. A kernel two-sample test. *J. Mach. Learn. Res.* **2012**, *13*, 723–773.
14. Villani, C. *Optimal Transport: OLD and New*; Springer: Berlin/Heidelberg, Germany, 2009; Volume 338.
15. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Adv. Neural Inf. Process. Syst.* **2017**, *30*. [CrossRef]
16. Theis, L.; Oord, A.v.d.; Bethge, M. A note on the evaluation of generative models. *arXiv* **2015**, arXiv:1511.01844.
17. Bińkowski, M.; Sutherland, D.J.; Arbel, M.; Gretton, A. Demystifying mmd gans. *arXiv* **2018**, arXiv:1801.01401.
18. Xu, Q.; Huang, G.; Yuan, Y.; Guo, C.; Sun, Y.; Wu, F.; Weinberger, K. An empirical study on evaluation metrics of generative adversarial networks. *arXiv* **2018**, arXiv:1806.07755.
19. Müller, A. Integral probability metrics and their generating classes of functions. *Adv. Appl. Probab.* **1997**, *29*, 429–443. [CrossRef]
20. Ramdas, A.; Trillos, N.G.; Cuturi, M. On wasserstein two-sample testing and related families of nonparametric tests. *Entropy* **2017**, *19*, 47. [CrossRef]
21. Genevay, A.; Peyré, G.; Cuturi, M. Learning generative models with sinkhorn divergences. In Proceedings of the International Conference on Artificial Intelligence and Statistics, Playa Blanca, Lanzarote, Spain, 9–11 April 2018; pp. 1608–1617.
22. Benjamini, Y.; Yekutieli, D. The control of the false discovery rate in multiple testing under dependency. *Ann. Stat.* **2001**, *29*, 1165–1188. [CrossRef]
23. Bounliphone, W.; Belilovsky, E.; Blaschko, M.B.; Antonoglou, I.; Gretton, A. A test of relative similarity for model selection in generative models. *arXiv* **2015**, arXiv:1511.04581.
24. Hoeffding, W. A Class of Statistics with Asymptotically Normal Distribution. *Ann. Math. Stat.* **1948**, 293–325. [CrossRef]
25. Gao, R.; Liu, F.; Zhang, J.; Han, B.; Liu, T.; Niu, G.; Sugiyama, M. Maximum mean discrepancy test is aware of adversarial attacks. In Proceedings of the International Conference on Machine Learning, Virtual, 18–24 July 2021; pp. 3564–3575.
26. Chen, X.; Yang, Y.; Li, Y. Augmented Sliced Wasserstein Distances. *arXiv* **2020**, arXiv:2006.08812.
27. Deshpande, I.; Hu, Y.T.; Sun, R.; Pyrros, A.; Siddiqui, N.; Koyejo, S.; Zhao, Z.; Forsyth, D.; Schwing, A.G. Max-Sliced Wasserstein distance and its use for GANs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 10648–10656.
28. Deshpande, I.; Zhang, Z.; Schwing, A.G. Generative modeling using the sliced wasserstein distance. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3483–3491.
29. Kolouri, S.; Nadjahi, K.; Simsekli, U.; Badeau, R.; Rohde, G. Generalized sliced wasserstein distances. In Proceedings of the Annual Conference on Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 261–272.
30. Nguyen, K.; Ho, N.; Pham, T.; Bui, H. Distributional sliced-Wasserstein and applications to generative modeling. *arXiv* **2020**, arXiv:2002.07367.