Learning Implicit Generative Models Using Differentiable Graph Tests

Josip Djolonga
Department of Computer Science
ETH Zurich
josipd@inf.ethz.ch

Andreas Krause
Department of Computer Science
ETH Zurich
krausea@ethz.ch

Abstract

Recently, there has been a growing interest in the problem of learning rich implicit models — those from which we can sample, but can not evaluate their density. One strategy of devising a loss function is through the statistics of two sample tests — if we can fool a statistical test, the learned distribution should be a good model of the true data. However, not all tests can easily fit into this framework, as they might not be differentiable with respect to the data points, and hence with respect to the parameters of the implicit model. Motivated by this problem, in this paper we show how two such classical tests, the Friedman-Rafsky and k-nearest neighbour tests, can be effectively smoothed using ideas from undirected graphical models – the matrix tree theorem and cardinality potentials.

1 Introduction

The main goal for our work is that of learning *implicit models*, i.e., those from which we can easily sample, but can not evaluate their density. Formally, we can generate a sample from an implicit distribution Q by first drawing $\mathbf{z}$ from some known and fixed distribution Q_0, typically Gaussian or uniform, and then passing it through some *differentiable* function $f_{\boldsymbol{\theta}}$ parametrized by some vector $\boldsymbol{\theta}$ to generate $\mathbf{x} = f_{\boldsymbol{\theta}}(\mathbf{z}) \sim Q$. The goal is then to optimize the parameters $\boldsymbol{\theta}$ of the mapping $f_{\boldsymbol{\theta}}$ so that Q is as close as possible to some target distribution P, which we can access only via iid samples. The approach that we undertake in this paper is that of defeating two-sample tests. These tests operate in the following setting — given two sets of iid samples, $X_1 = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_{n_1}\}$ from P, and $X_2 = \{\mathbf{x}_{n_1+1}, \mathbf{x}_{n_1+2}, \dots, \mathbf{x}_{n_1+n_2}\}$ from Q, we have to distinguish between the hypotheses

$$H_0 \colon P = Q \quad \text{vs} \quad H_1 \colon P \neq Q.$$

The tests that we consider start by defining a function $T \colon (\mathbb{R}^d)^{n_1} \times (\mathbb{R}^d)^{n_2} \to \mathbb{R}$ that should result in a *low* value if the two samples come from different distributions. Then, the hypothesis H_0 is rejected at significance level $\alpha \in [0, 1]$ if $T(X_1, X_2)$ is lower than some threshold t_α, which is computed using a permutation test, as explained in §2. Going back to the original problem, one intuitive approach would be to maximize the expected statistic $\mathbb{E}_{\mathbf{x}_i \sim P, \mathbf{z}_i \sim Q_0}[T(\{\mathbf{x}_i\}_{i=1}^{n_1}, \{f_{\boldsymbol{\theta}}(\mathbf{z}_i)\}_{i=n_1}^{n_1+n_2})]$ using stochastic optimization over the parameters of the mapping $f_{\boldsymbol{\theta}}$. However, this requires the availability of the derivatives $\partial T / \partial \mathbf{x}_i$, which is unfortunately not always possible. For example, the Friedman-Rafsky (FR) and k-nearest neighbours (k-NN) tests, which have very desirable statistical properties (including consistency and convergence of their statistics to f-divergences), can not be cast in the above framework as they use the *output* of a combinatorial optimization problem. Our main contribution is the development of differentiable versions of these tests that remedy the above problem by smoothing their statistics. We moreover show, similarly to these classical tests, that our tests are asymptotically normal under certain conditions, and derive the corresponding t-statistic, which can be evaluated with minimal additional complexity. Our smoothed tests can have more power over their classical variants, as we showcase with numerical experiments. Finally, we experimentally learn implicit models in §5.

Related work. The problem of two-sample testing for distributional equality has received significant interest in statistics [1, 2, 3, 4]. The idea of using statistical two-sample tests to learn implicit models has been used in [5, 6]. In [7] the authors also suggest a t-statistic for learning implicit models. The classical versions of the graph test we consider have been originally introduced in [1, 2]. For an overview of various approaches to learning implicit models we direct the reader to [8].

2 Classical Graph Tests

Let us start by introducing some notation. For any set $X = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$ of points in $\mathbb{R}^d$, we will denote by $\mathcal{G}(X) = (X, E)$ the *complete directed graph*[1] defined over the vertex set X with edges E. We will moreover weight this graph using a function $d\colon \mathbb{R}^d \times \mathbb{R}^d \to [0, \infty)$, e.g. a natural choice would be $d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|$. Similarly, we will use $d(e)$ for the weight of the edge e under $d(\cdot, \cdot)$. For any labelling of the vertices $\pi : X \to \{1, 2\}$, and any edge $e \in E$ with adjacent vertices i and j we define[2] $\Delta_\pi(e) = [\![\pi(i) \neq \pi(j)]\!]$, i.e., $\Delta_\pi(e)$ indicates if its end points of e have *different* labels under π. Remember that we are given n_1 points $X_1 = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n_1}\}$ from P, and n_2 points $X_2 = \{\mathbf{x}_{n_1+1}, \mathbf{x}_{n_1+2}, \ldots, \mathbf{x}_{n_1+n_2}\}$ from Q. In the remaining of the paper we will use $n = n_1 + n_2$ for the total number of points. The tests are based on the following four-step strategy.

(i) Pool the samples X_1 and X_2 together into $X = X_1 \cup X_2 = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n_1+n_2}\}$, and create the graph $\mathcal{G}(X)$. Define the mapping $\pi^*\colon X \to \{1, 2\}$ evaluating to 1 on X_1 and to 2 on X_2.

(ii) Using some well-defined algorithm $\mathcal{A}$ choose a subset $U^* = \mathcal{A}(\mathcal{G}(X))$ of the *edges* of this graph with the underlying motivation that it defines some neighbourhood structure.

(iii) Count how many edges in U^* connect points from X_1 with points from X_2, i.e., compute the statistic $T_{\pi^*}(U^*) = \sum_{e \in U^*} \Delta_{\pi^*}(e)$.

(iv) Reject H_0 for small values of $T_{\pi^*}(U^*)$.

These tests condition on the data and are executed as permutation tests, so that the critical value in step (iv) is computed using the quantiles of $\mathbb{E}_{\pi \sim H_0} T_\pi(U^*)$, where $\pi\colon X \to \{1, 2\}$ is drawn uniformly at random from the set of $\binom{n_1+n_2}{n_1}$ labellings that map exactly n_1 points from X to 1. Formally, the p-value is given as $\mathbb{E}_{\pi \sim H_0}[[\![T_{\pi^*}(U^*) \geq T_\pi(U^*)]\!]]$.

Friedman-Rafsky (FR) [2]. This test uses the minimum-spanning tree (MST) of $\mathcal{G}(X)$ as the neighbourhood structure U^*. If we use $d(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|$, Henze and Penrose [9] have proven an asymptotic limit for the test statistic, which after some algebraic manipulation, as noted in [10], can be shown that $1 - T_{\pi^*}(U^*)\frac{n_1+n_2}{2n_1n_2}$ converges almost surely to the following f-divergence [11] $D_\alpha^{\text{FR}}(P \,\|\, Q) = \frac{1}{4\alpha(1-\alpha)} \int \frac{(\alpha p(\mathbf{x}) - (1-\alpha) q(\mathbf{x}))^2}{\alpha p(\mathbf{x}) + (1-\alpha) q(\mathbf{x})} d\mathbf{x} - \frac{(2\alpha-1)^2}{4\alpha(1-\alpha)}$. In [10] it is also noted that if $n_1 = n_2$, then $D_{1/2}$ is equal to $2 \int \frac{(p(\mathbf{x}) - q(\mathbf{x}))^2}{p(\mathbf{x}) + q(\mathbf{x})} d\mathbf{x}$, which is known as the symmetric χ^2 divergence.

k-nearest-neighbours (k-NN) [1]. Maybe the most intuitive way to construct a neighbourhood structure is to connect each point $\mathbf{x}_j \in X$ to its k nearest neighbours. Specifically, we add the edge $\mathbf{x}_i \to \mathbf{x}_j$ to U^* iff $\mathbf{x}_i$ is one of the k closest neighbours of $\mathbf{x}_j$. If one uses the Euclidean norm, then the asymptotic distribution and the consistency of the test have been proven by Schilling [12]. Namely, it is known that $\frac{T_{\pi^*}(U^*)}{(n_1+n_2)k}$ converges in probability to $D_\alpha^{\text{NN}}(P \,\|\, Q) \equiv \int \frac{\alpha^2 p^2(\mathbf{x}) + (1-\alpha)^2 q^2(\mathbf{x})}{\alpha p(\mathbf{x}) + (1-\alpha) q(\mathbf{x})} d\mathbf{x}$.

3 Differentiable Graph Tests

Unfortunately, the tests from the previous section can not be used to train implicit models because the derivatives $\partial T / \partial \mathbf{x}_i$ are either zero or do not exist, as T takes on finitely many values. The strategy that we undertake is to *smooth* them by relaxing them to expectations in natural probabilistic models. To motivate the models we will introduce, note that for both tests the optimal neighbourhood is the solution to the following optimization problem

$$U^* = \operatorname*{arg\,min}_{U \subseteq E} \sum_{e \in U} d(e) \text{ s.t. } \nu(U) = 1, \tag{1}$$

[1] For the FR test we will arbitrarily choose one of the two edges for each pair of nodes.

[2] We use the Iverson bracket $[\![S]\!]$ that evaluates to 1 if S is true and 0 otherwise.

where $\nu\colon 2^E \to \{0,1\}$ indicates if the set of edges is *valid*, i.e., if every vertex has exactly k neighbours in the k-NN case, or if the set of edges forms a poly-tree in the MST case. Moreover, note that once we fix n_1 and n_2, the optimization problem (1) depends only on the edge weights $d(e)$, which we will concatenate in an arbitrary order and store in the vector $\mathbf{d} \in \mathbb{R}^{|E|}$. We want to design a probability distribution over U that focuses on those configurations U that are both feasible and have a low cost for problem (1). One such natural choice is the following Gibbs measure

$$P(U \mid \mathbf{d}/\lambda) = e^{-\sum_{e\in U} d(e)/\lambda - A(-\mathbf{d}/\lambda)}\nu(U), \tag{2}$$

where λ is the so-called temperature parameter, and $A(-\mathbf{d}/\lambda)$ is the log-partition function that ensures that the distribution is normalized. Note that U^* is a MAP configuration of distribution (2), and the distribution will concentrate on the MAP configurations as $\lambda \to 0$. Once we have fixed the model, the strategy is clear — replace the statistic $T_{\pi^*}(U^*)$ with its expectation $\mathbb{E}_U[T_{\pi^*}(U)]$, which results in the following smooth statistic

$$T_{\pi *}(U^*) \longrightarrow T^{\lambda}_{\pi^*} \equiv \mathbb{E}_{U\sim P(\cdot|\mathbf{d},\lambda)}[T_{\pi^*}(U)] = \sum_{e\in E} \Delta_{\pi^*}(e)\boldsymbol{\mu}(\mathbf{d}/\lambda)_e,$$

where $\boldsymbol{\mu}(\mathbf{d}/\lambda)$ are the marginal probabilities of the edges, i.e., $[\boldsymbol{\mu}(\mathbf{d}/\lambda)]_e = \mathbb{E}_{P(U|\mathbf{d}/\lambda)}[[\![e \in U]\!]]$. Hence, we can compute the statistic as long as we can perform inference in (2). To compute its derivatives we can use the fact that (2) is a member of the exponential family. Namely, leveraging the classical properties of log-partition functions [13, Prop. 3.1], we obtain the following identities $\boldsymbol{\mu}(\mathbf{d}/\lambda) = \nabla A(-\mathbf{d}/\lambda)$ and $\frac{\partial \boldsymbol{\mu}(\mathbf{d}/\lambda)_e}{\partial \boldsymbol{\mu}(\mathbf{d}/\lambda)_{e'}} = \mathbb{E}_{P(U|\mathbf{d}/\lambda)}[[\![\{e,e'\} \subseteq U]\!]] - \boldsymbol{\mu}(\mathbf{d}/\lambda)_e\boldsymbol{\mu}(\mathbf{d}/\lambda)_{e'}$. Thus, if we can compute the second-order moments of (2), we get the smoothed statistic and its derivative.

A smooth p-value. Even though one can use the smoothed test statistic $T^{\lambda}_{\pi^*}$ as an objective when learning implicit models, it does not necessarily mean that lower values of this statistic result in higher p-values. Remember that to compute a p-value, one has to run a permutation test by computing quantiles of T^{λ}_{π} under $\pi \sim H_0$. However, as this procedure is not smooth and costly to compute, we suggest, similarly as [2, 9, 12], the following t-statistic $t^{\lambda}_{\pi^*} = \frac{T^{\lambda}_{\pi^*} - \mathbb{E}_{\pi\sim H_0}[T^{\lambda}_{\pi}]}{\sqrt{\mathbb{V}_{\pi\sim H_0}[T^{\lambda}_{\pi}]}}$, which, as shown in the appendix, can be easily computed, and is moreover asymptotically normal under certain conditions.

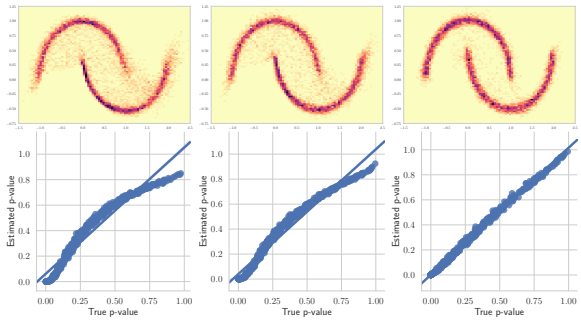
4 The Differentiable k-NN and FR Tests

In this section, we discuss these two tests in more detail and show how to efficiently compute their statistics, by showing how to do inference in the corresponding models. We would like to stress that, in the learning setting that we consider n refers to the number of data-points in a *mini-batch*.

k-NN. The constraint $\nu(\cdot)$ in this case requires the total number of edges in U incoming at each node to be exactly k. First, note that the problem completely *separates* per node, i.e., the marginals of edges with different target vertices are independent. Formally, if we denote by U_i the set of edges incoming at vertex i, then U_i and U_j are independent for $i \neq j$. Hence, for each node i *separately*, we have to perform inference in $P(U_i) \propto \exp(-\sum_{j\in U_i} d(\mathbf{x}_i, \mathbf{x}_j)/\lambda)[\![|U_i| = k]\!]$, which is a special case of the cardinality potentials considered by Tarlow et al. [14] and Swersky et al. [15], who show how to compute *all marginals* in time $O(nk)$. Moreover, as marginalization requires only simple operations, we can compute the derivatives with any automatic differentiation software, and we thus do not provide formulas for the second-order moments. As a concrete example, let us work out the simplest case — the k-NN test with $k = 1$. In this case, the smoothed statistic reduces to

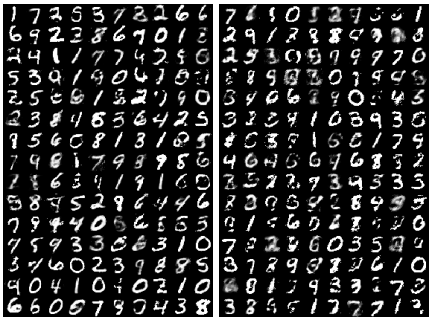
$$T^{\lambda}_{\pi^*}(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ \pi^*(i)\neq\pi^*(j)}}^{n} \texttt{softmax}(-\otimes_{l\neq i} \|\mathbf{x}_i - \mathbf{x}_l\|/\lambda).$$

We thus want to maximize the number of *incorrect* predictions if we are to estimate the label $\pi(i)$ from $\mathbf{x}_i$ using a soft 1-nearest neighbour approach. Furthermore, we can also make a clear connection between the smooth 1-NN test and neighbourhood component analysis (NCA) [16]. Namely, we can see NCA as learning a mapping $h\colon \mathbf{x} \to A\mathbf{x}$ so that the test *distinguishes* (by minimizing $T^{\lambda}_{\pi^*}$) the two samples as best as possible after applying h on them. The extension of NCA to k-NN [17] can be also seen as minimizing the test statistic for a particular instance of their loss function.



(a) $\lambda = 10$. (b) $\lambda = 1$. (c) $\lambda = 0.05$.

Figure 1: Effects of varying λ on the learned model and the normality of the statistic on 1-NN. With decreasing λ we get closer to normality, and the learned distribution also improves.



(a) 1-NN with $\lambda = 10$ and $n_1 = 512$. (b) FR $\lambda = 20$ and $n_1 = 128$.

Figure 2: Different models trained on MNIST.

Friedman-Rafsky. The model that we have to perform inference in for this test seems intractable as the constraint has the form $\nu(U) = [\![U \text{ forms a spanning tree}]\!]$. Fortunately, Lyons [18] has shown that the above model is a determinantal point process (DPP), so that marginalization can be done easily in $O(n^3)$ time. To speed up this computation we can leverage the existing theory on fast solvers of Laplacian systems. Let us first create from $\mathcal{G}(X)$ the graph $e^{\mathcal{G}}(X)$ that has the same structure as $\mathcal{G}(X)$, but with edge weights $e^{-d(e)/\lambda}$ instead of $d(e)$. In $e^{\mathcal{G}}(X)$, the marginals $\boldsymbol{\mu}_e$ are also known as *effective resistances* [19]. Spielman and Srivastava [20] provide a method to compute *all* marginals at once in time that is $\tilde{O}(rn^2/\varepsilon^2)$, where ε is the desired precision and $r = \frac{1}{\lambda}(\max_e d(e) - \min_e d(e))$. As an extra benefit, this connection provides an alternative interpretation of the smoothed FR test. Namely, assume that we want to create a *spectral sparsifier* [21] of $e^{\mathcal{G}}(X)$, which should contain *significantly less* edges, but be a good summary of the graph by having a similar spectrum. Spielman and Srivastava [20] provide a strategy to create such a sparsifier by sampling edges randomly, where edge e is sampled proportional to μ_e. Hence, by optimizing $T^{\lambda}_{\pi^*}$ we are encouraging the sparsifier of $e^{\mathcal{G}}(X)$ to have as many edges as possible connecting points from X_1 with points from X_2.

5 Experiments

For the k-NN test, we have adapted the code accompanying [15]. We used a 10 dimensional normal as Q_0, drew samples of equal size $n_1 = n_2$, and used $d(\mathbf{x}, \mathbf{x}') = \|\mathbf{x} - \mathbf{x}'\|_2$ as a weighting function.

Statistical power. In the appendix we provide experiments that compare the power against several other tests under various alternatives H_1. We have found that smoothing can significantly improve the power of the test, and our tests are moreover better than competing two sample tests.

Learning. We stochastically maximize t_{π^*} using Adam [22]. The first experiment we perform, with the goal of understanding the effects of λ, is on the toy *two moons* dataset [23]. We show the results in Section 5. From the second row, showing the estimated p-value versus the correct one (from 1000 random permutations) at several points during training, we can indeed see that the permutation null gets closer to normality as λ decreases. Most importantly, note that the relationship is monotone, so that we would expect the optimization to not be significantly harmed if we use the approximation. Qualitatively, we can observe that the solutions have the general structure of P, and that they improve as we decrease λ — the symmetry is better captured and the two moons get better separated.

We have also trained several models on MNIST [24], which we present in Figure 2. Despite the high (784) dimensional data and the fact that we use the distance directly on the pixels, the models generate digits that look mostly realistic and are competitive with obtained using MMD [5, 6].

6 Conclusion

We have developed smooth two-sample graph tests that can be used for learning implicit models, analyzed them theoretically, and showcased their benefits with numerical experiments.

References

[1] Jerome H Friedman and Lawrence C Rafsky. "Graph-theoretic measures of multivariate association and prediction". *Annals of Statistics* (1983), pp. 377–391.

[2] Jerome H Friedman and Lawrence C Rafsky. "Multivariate generalizations of the Wald-Wolfowitz and Smirnov two-sample tests". *Annals of Statistics* (1979), pp. 697–717.

[3] Gábor J Székely and Maria L Rizzo. "Energy statistics: A class of statistics based on distances". *Journal of Statistical Planning and Inference* 143.8 (2013), pp. 1249–1272.

[4] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. "A kernel two-sample test". *Journal of Machine Learning Research* 13.Mar (2012), pp. 723–773.

[5] Yujia Li, Kevin Swersky, and Rich Zemel. "Generative moment matching networks". *International Conference on Machine Learning (ICML)*. 2015.

[6] Gintare Karolina Dziugaite, Daniel M. Roy, and Zoubin Ghahramani. "Training Generative Neural Networks via Maximum Mean Discrepancy Optimization". *Uncertainty in Artificial Intelligence (UAI)*. 2015.

[7] Dougal J Sutherland, Hsiao-Yu Tung, Heiko Strathmann, Soumyajit De, Aaditya Ramdas, Alex Smola, and Arthur Gretton. "Generative models and model criticism via optimized maximum mean discrepancy". *International Conference on Learning Representations (ICLR)*. 2016.

[8] Shakir Mohamed and Balaji Lakshminarayanan. "Learning in implicit generative models". *arXiv preprint arXiv:1610.03483* (2016).

[9] Norbert Henze and Mathew D Penrose. "On the multivariate runs test". *Annals of Statistics* (1999), pp. 290–298.

[10] Visar Berisha and Alfred O Hero. "Empirical non-parametric estimation of the Fisher information". *IEEE Signal Processing Letters* 22.7 (2015), pp. 988–992.

[11] Syed Mumtaz Ali and Samuel D Silvey. "A general class of coefficients of divergence of one distribution from another". *Journal of the Royal Statistical Society. Series B (Methodological)* (1966), pp. 131–142.

[12] Mark F Schilling. "Multivariate two-sample tests based on nearest neighbors". *Journal of the American Statistical Association* 81.395 (1986), pp. 799–806.

[13] Martin J Wainwright and Michael I Jordan. "Graphical models, exponential families, and variational inference". *Foundations and Trends® in Machine Learning* 1.1-2 (2008).

[14] Daniel Tarlow, Kevin Swersky, Richard S Zemel, Ryan P Adams, and Brendan J Frey. "Fast Exact Inference for Recursive Cardinality Models". *Uncertainty in Artificial Intelligence (UAI)*. 2012.

[15] Kevin Swersky, Ilya Sutskever, Daniel Tarlow, Richard S Zemel, Ruslan R Salakhutdinov, and Ryan P Adams. "Cardinality Restricted Boltzmann Machines". *Advances in Neural Information Processing Systems (NIPS)*. 2012, pp. 3293–3301.

[16] Jacob Goldberger, Geoffrey E Hinton, Sam T Roweis, and Ruslan R Salakhutdinov. "Neighbourhood components analysis". *Advances in Neural Information Processing Systems (NIPS)*. 2005, pp. 513–520.

[17] Daniel Tarlow, Kevin Swersky, Laurent Charlin, Ilya Sutskever, and Rich Zemel. "Stochastic k-neighborhood selection for supervised and unsupervised learning". *International Conference on Machine Learning*. 2013, pp. 199–207.

[18] Russell Lyons. "Determinantal probability measures". *Publications mathématiques de l'IHÉS* 98.1 (2003), pp. 167–212.

[19] Ashok K Chandra, Prabhakar Raghavan, Walter L Ruzzo, Roman Smolensky, and Prasoon Tiwari. "The electrical resistance of a graph captures its commute and cover times". *Computational Complexity* 6.4 (1996), pp. 312–340.

[20] Daniel A Spielman and Nikhil Srivastava. "Graph sparsification by effective resistances". *SIAM Journal on Computing* 40.6 (2011), pp. 1913–1926.

[21] Daniel A Spielman and Shang-Hua Teng. "Spectral sparsification of graphs". *SIAM Journal on Computing* 40.4 (2011), pp. 981–1025.

[22] Diederik Kingma and Jimmy Ba. "Adam: A method for stochastic optimization". *International Conference on Learning Representations (ICLR)*. 2015.

[23] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. "Scikit-learn: Machine Learning in Python". *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[24] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. "Gradient-based learning applied to document recognition". *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[25] Henry E Daniels. "The relation between measures of correlation in the universe of sample permutations". *Biometrika* 33.2 (1944), pp. 129–135.

[26] AD Barbour and GK Eagleson. "Random association of symmetric arrays". *Stochastic Analysis and Applications* 4.3 (1986), pp. 239–281.

[27] Sören Sonnenburg, Sebastian Henschel, Christian Widmer, Jonas Behr, Alexander Zien, Fabio de Bona, Alexander Binder, Christian Gehl, Vojtěch Franc, et al. "The SHOGUN machine learning toolbox". *Journal of Machine Learning Research* 11 (2010), pp. 1799–1802.

[28] Aaditya Ramdas, Sashank Jakkam Reddi, Barnabás Póczos, Aarti Singh, and Larry A Wasserman. "On the Decreasing Power of Kernel and Distance Based Nonparametric Hypothesis Tests in High Dimensions". *AAAI*. 2015.

[29] Bhaswar B Bhattacharya. "Power of graph-based two-sample tests". *arXiv preprint arXiv:1508.07530* (2015).

A Computation of the t-statistic

Before we show to compute the first two moments under H_0, we need to define the matrix Π holding the second moments of the variables $\Delta_\pi(e)$.

Lemma 1 ([2]). *The matrix $\Pi \in \mathbb{R}^{|E|\times|E|}$ with entries $\Pi_{e,e'} = \mathbb{E}_{\pi\sim H_0}[\Delta_\pi(e)\Delta_\pi(e')]$ is equal to*

$$\Pi_{e,e'} = \begin{cases} \frac{2n_1n_2}{n(n-1)} & \text{if } \delta(e) = \delta(e'), \text{ or} \\ \frac{n_1n_2}{n(n-1)} & \text{if } |\delta(e)\cap\delta(e')| = 1, \text{ or} \\ \frac{4n_1n_2(n_1-1)(n_2-1)}{n(n-1)(n-2)(n-3)} & \text{if } \delta(e)\cap\delta(e') = \emptyset, \end{cases}$$

where $\delta(e)$ is the set of vertices incident to the edge $e \in E$.

Theorem 1. *Assume that all valid configurations U satisfy $|U| = m$, i.e. that $\nu(U) \neq 0$ implies $|U| = m$.[3] Then, the first two moments of the statistic under H_0 are*

$$\mathbb{E}_{\pi\sim H_0}[T^\lambda_{\pi^*}] = 2mn_1n_2/n(n-1), \text{ and}$$
$$\mathbb{V}_{\pi\sim H_0}[T^\lambda_{\pi^*}] = \boldsymbol{\mu}(\mathbf{d}/\lambda)^T\Pi\boldsymbol{\mu}(\mathbf{d}/\lambda) - 4\frac{n_1^2n_2^2}{n^2(n-1)^2}m^2.$$

Proof. The expectation of the statistic under H_0 is (when π is a uniformly random labelling)

$$\sum_{e\in E}\mu(\mathbf{d}/\lambda)_e \underbrace{\mathbb{E}_\pi[\Delta_\pi(e)]}_{2n_1n_2/n(n-1)} = 2mn_1n_2/n(n-1),$$

where the inner expectation $\mathbb{E}_\pi[\Delta_\pi(e)]$ has been computed by [2]. We can also easily compute the variance as

$$\sum_{e,e'\in E}\mathrm{Cov}_{\pi\sim H_0}[\mu_e\Delta_\pi(e), \mu_{e'}\Delta_\pi(e')] = \sum_{e,e'\in E}\mu_e\mu_{e'}\underbrace{\mathbb{E}_{\pi\sim H_0}[\Delta_\pi(e)\Delta_\pi(e')]}_{\Pi_{e,e'}} - \underbrace{\frac{4n_1^2n_2^2}{n^2(n-1)^2}m^2}_{(\mathbb{E}_{\pi\sim H_0}[T^\lambda_{\pi^*}])^2}. \tag{3}$$

□

While the computation of the mean is trivial, it seems that the computation of the variance needs $O(|E|^2)$ operations. However, we can simplify its computation to $O(|E|)$ using the following result.

Lemma 2. *Define $\chi_1 = \frac{n_1n_2}{n(n-1)}$ and $\chi_2 = \frac{4(n_1-1)(n_2-1)}{(n-2)(n-3)}$. The variance can be then computed as*

$$\sigma^2 = \chi_1(1-\chi_2)\sum_v(\sum_{e\in\delta(v)}\mu_e)^2 + \chi_1\chi_2\sum_{e\parallel e'}\mu_e\mu_{e'} + \chi_1(\chi_2 - 4\chi_1)m^2,$$

where $\sum_{e\parallel e}$ sums over all pairs of parallel edges, i.e., those connecting the same end-points.

Proof. We can split the sum in the variance formula over all edge pairs into three groups as follows

$$\sum_e\sum_{e'\sim e}\underbrace{\frac{n_1n_2}{n(n-1)}}_{\chi_1}\mu_e\mu_{e'} + \sum_e\sum_{e'\perp e}\underbrace{\frac{4n_1n_2(n_1-1)(n_2-1)}{n(n-1)(n-2)(n-3)}}_{\chi_1\chi_2}\mu_e\mu_{e'} + \sum_e\underbrace{\frac{n_1n_2}{n(n-1)}}_{\chi_1}(\mu_e^2 + \mu_e\mu_{\overline{e}}), \tag{4}$$

where $\sum_{e'\sim e}$ sums over all edges e' that share at least one vertex with e, and $\sum_{e'\perp e}$ sums over those edges that share no vertex with e, and $\overline{e}$ denote the *reverse* edge of e (if it exist, zero otherwise). Note that each term $\mu_e\mu_{e'}$ appears twice if $e \neq e'$, as in the formula for the variance (3). Moreover, note that if $\delta(e) = \delta(e')$, then in the above formula the term $\mu_e\mu_{e'}$ (same for $\mu_{e'}\mu_e$) gets multiplied by $2\chi_1 = \Pi_{e,e'}$, as it appears in both the first and the third term. Given that assumption that $|U| = m$ under $\nu(\cdot)$, we also know that

$$m^2 = (\sum_e\mu_e)^2 = \sum_e\sum_{e'}\mu_e\mu_{e'} = \sum_e\sum_{e'\sim e}\mu_e\mu_{e'} + \sum_e\sum_{e'\perp e}\mu_e\mu_{e'},$$

[3] Note that we have $m = kn$ for k-NN and $m = n-1$ for FR.

so that eq. (4) can be simplified to

$$\chi_1 \sum_e \sum_{e' \sim e} \mu_e \mu_{e'} + \chi_1 \chi_2 (m^2 - \sum_e \sum_{e' \sim e} \mu_e \mu_{e'}) + \chi_1 \sum_e (\mu_e^2 + \mu_e \mu_{\overline{e}}),$$

which be simplified to

$$\chi_1 (1 - \chi_2) \sum_e \sum_{e' \sim e} \mu_e \mu_{e'} + \chi_1 \sum_e (\mu_e^2 + \mu_e \mu_{\overline{e}}) + \chi_1 \chi_2 m^2.$$

Now the result follows by observing that

$$\sum_v (\sum_{e \in \delta(v)} \mu_e)^2 = \sum_e \sum_{e' \sim e} \mu_e \mu_{e'} + \sum_e \mu_e^2 + \sum_e \mu_e \mu_{\overline{e}}.$$

To understand why this holds, let us count how many times each term $\mu_e \mu_{e'}$ appears on both sides of the equality if we expand the lhs. If $e \neq e'$ and they share exactly one vertex, then the lhs will have two $\mu_e \mu_{e'}$ terms, as μ_e and $\mu_{e'}$ will be multiplied only at the term corresponding to the shared vertex. On the other hand, if $e = e'$ we will again have two $\mu_e \mu_{e'} = \mu_e^2$ terms, as we get one contribution from each end-point of e. Finally, if $e' = \overline{e}$, we have a total of four $\mu_e \mu_{e'}$ terms, as we get two $\mu_e \mu_{e'}$ from each end-point. Thus, eq. (4) is equal to

$$\chi_1 (1 - \chi_2) \big(\sum_v (\sum_{e \in \delta(v)} \mu_e)^2 - \sum_e \mu_e^2 - \sum_e \mu_e \mu_{\overline{e}} \big) + \chi_1 \sum_e (\mu_e^2 + \mu_e \mu_{\overline{e}}) + \chi_1 \chi_2 m^2.$$

Finally, if we subtract $4\chi_1^2 m^2$ and simplify the expression we have

$$\chi_1 (1 - \chi_2) \sum_v (\sum_{e \in \delta(v)} \mu_e)^2 + \chi_1 \chi_2 \sum_e \mu_e^2 + \chi_1 \chi_2 \sum_e \mu_e \mu_{\overline{e}} + \chi_1 (\chi_2 - 4\chi_1) m^2,$$

which is exactly what is claimed in the theorem, if we observe that e and $\overline{e}$ are the only edges parallel to e. □

B Approximate normality

To better motivate the use of a t-statistic, we can, similarly to the arguments in [2, 9, 12], show that it is is close to a normal distribution by casting it as a *generalized correlation coefficient* [25, 1]. Namely, these are tests whose statistics are the form form $\kappa = \sum_{i=1}^n \sum_{j=1}^n \overline{\mu}_{i,j} b_{i,j}$, and whose critical values are computed using the distribution of $\sum_{i=1}^n \sum_{j=1}^n \overline{\mu}_{i,j} b_{\pi(i),\pi(j)}$, where π is a random permutation on $\{1, 2, \ldots, n\}$. It is easily seen that we can fit the suggested tests in this framework if we set $\overline{\mu}_{i,j} = \frac{1}{2}(\boldsymbol{\mu}(\mathbf{d}/\lambda)_{i \to j} + \boldsymbol{\mu}(\mathbf{d}/\lambda)_{j \to i})$ and $b_{i,j} = \Delta_{\pi^*}(\{i, j\})$. Then, using the conditions of Barbour and Eagleson [26], we obtain the following bound on the deviation from normality.

Theorem 2. *Let $n_1/(n_1 + n_2) \to \alpha \in (0, 1)$, and define*

- $S_2 = \sum_{i,j,k} \overline{\mu}_{i,j} \overline{\mu}_{i,k}$*, i.e., the expected number of edges sharing a vertex,*
- $S_3 = \sum_{i,j,k,m} \overline{\mu}_{i,j} \overline{\mu}_{i,k} \overline{\mu}_{i,m}$*, i.e., the expected number of 3 stars, and*
- $L_4 = \sum_{i,j,k,m} \overline{\mu}_{i,j} \overline{\mu}_{j,k} \overline{\mu}_{k,m}$*, i.e., the expected number of paths with 4 nodes.*

Then, the Wasserstein distance between the permutation null $\mathbb{E}_{\pi \sim H_0}[T_\pi^\lambda(U^*)]$ and the standard normal is of order $O\big((nk^3 + kS_2 + S_3 + L_4)/\sigma^3\big)$.

Proof. Let us compute an upper bound on the quantities in [26].

$$a_1 = \frac{1}{n(n-1)} \sum_{i,j} \overline{\mu}_{i,j} = \frac{k}{n} \qquad b_1 = \frac{2}{n(n-1)} n_2 n_1 = \Theta(1)$$

$$a_2 = \frac{1}{n(n-1)(n-2)} \underbrace{\sum_{i,j,k} \overline{\mu}_{i,j}\overline{\mu}_{i,k}}_{S_2} \qquad b_2 = \frac{n_2 n_1^2 + n_1 n_2^2}{n(n-1)(n-2)} = \Theta(1)$$

$$a_3 = \frac{1}{n(n-1)(n-2)(n-3)} \underbrace{\sum_{i,j,k,m} \overline{\mu}_{i,j}\overline{\mu}_{i,k}\overline{\mu}_{i,m}}_{S_3} \qquad b_3 = \frac{n_2 n_1^3 + n_1 n_2^3}{n(n-1)(n-2)(n-3)} = \Theta(1)$$

$$a_4 = \frac{1}{n(n-1)(n-2)(n-3)} \underbrace{\sum_{i,j,k,m} \overline{\mu}_{k,i}\overline{\mu}_{i,j}\overline{\mu}_{j,m}}_{L_4} \qquad b_4 = 2\frac{n_2^2 n_1^2}{n(n-1)(n-2)(n-3)} = \Theta(1)$$

$$a_5 = \frac{1}{n(n-1)(n-2)} \sum_{i,j,k} \overline{\mu}_{i,j}^2 \overline{\mu}_{i,k} = O(a_2) \qquad b_5 = b_2$$

$$a_6 = \frac{1}{n(n-1)} \sum_{i,j} \overline{\mu}_{i,j}^3 = O(a_1) \qquad b_6 = b_1$$

$$a_7 = \frac{1}{n(n-1)(n-2)} \sum_{i,j,k,m} \overline{\mu}_{i,j}\overline{\mu}_{i,k}\overline{\mu}_{j,k} \qquad b_7 = \frac{n_2 n_1 n_2 + n_1 n_2 n_1}{n(n-1)(n-2)} = \Theta(1)$$

$$a_8 = \frac{1}{n(n-1)} \sum_{i,j} \overline{\mu}_{i,j}^2 = O(a_1) \qquad b_8 = b_1.$$

Then, the upper bound has the form

$$\frac{1}{\sigma^3}\big[n^4\big(\underbrace{a_1^3}_{k^3/n^3} + \underbrace{a_1 a_2}_{O(kS_2/n^4)} + \underbrace{a_3}_{O(S_3/n^4)} + \underbrace{a_4}_{O(L_4/n^4)}\big)\underbrace{(b_1^3 + b_1 b_2 + b_3 + b_4)}_{O(1)} + n^3\big(\underbrace{a_5}_{O(S_2/n^3)} + \underbrace{a_1 a_8}_{O(k^2/n^2)}\big)\underbrace{(b_5 + b_1 b_8)}_{O(1)} + n^2 \underbrace{a_6}_{O(k/n)}\underbrace{b_6}_{O(1)}\big],$$

which can be simplified to

$$O\big(\frac{1}{\sigma^3}\big[nk^3 + kS_2 + S_3 + L_4 + S_2 + nk^2 + k/n\big]\big) = O\big(\frac{1}{\sigma^3}(nk^3 + kS_2 + S_3 + L_4)\big),$$

which is what is claimed in the theorem. □

C Experiments

C.1 Power test

This this experiment we analyze the effect of the smoothing strength on the power of our differentiable tests. In addition to the classical FR and k-NN tests, we have considered the unbiased MMD test [4] with the squared exponential kernel $k(\mathbf{x}, \mathbf{x}') = e^{-\frac{1}{\lambda^2}\|\mathbf{x}-\mathbf{x}'\|^2}$ (as implemented in `Shogun` [27] using the code from [7]), and the energy test [3]. The problem that we consider, which is challenging in high dimensions, is that of differentiating the distribution $\mathcal{N}(\mathbf{0}, I)$ from $\mathcal{N}((\mu, 0, \ldots, 0), \mathtt{diag}(\sigma^2, 1, \ldots, 1))$. This setting was considered to be fair in [28], as the KL divergence between the distribution is constant irrespective of the dimension. To set the bandwidth of the MMD kernel (in addition to the median heuristic) we used the same strategy as in [28] by setting $\lambda = d^\gamma$ for varying $\gamma \in [0, 1]$. The results are presented in Figure 3, where we can observe that (i) our test have similar results with MMD for shift-alternatives, while performing significantly better for scale alternatives, and (ii) by varying the smoothing parameter we can significantly increase the power of the test. In the third column we present only the best performing MMD, while we present the remaining results in Figure 4. Note that we expect the power to go to zero as the dimension increases [29, 28].

C.2 MMD

C.3 Architecture

We have used the same architecture as in [5, 7], which using the modules from `PyTorch` can be written as follows.

```
nn.Sequential(
    nn.Linear(noise_dim, 64),
    nn.ReLU(),
    nn.Linear(64, 256),
    nn.ReLU(),
    nn.Linear(256, 256),
    nn.ReLU(),
    nn.Linear(256, 1024),
    nn.ReLU(),
    nn.Linear(1024, ambient_dim))
```

For MNIST we have also added a terminal `nn.Tanh` layer.

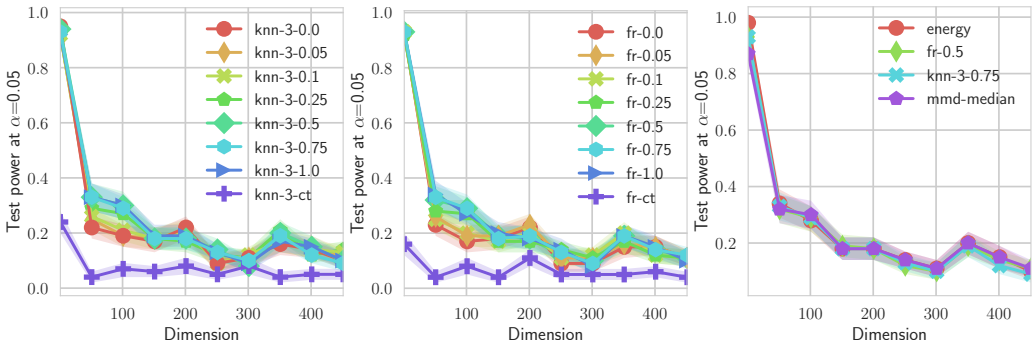
C.4 Data

We have used the MNIST data as packaged by `torchvision`, with the additional processing of scaling the output to $[-1, 1]$ as we are using a final `Tanh` layer. For the *two moons* data, we have used a noise level of 0.05.

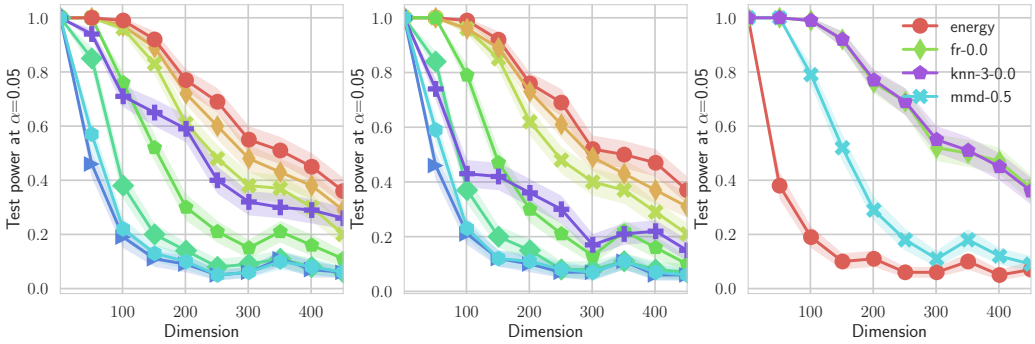
C.5 Optimization

All details are provided in the table below. In some cases we have optimized with a larger step for a number of epochs, and then reduced it for the remaining epochs — in the table below these are separated by commas.

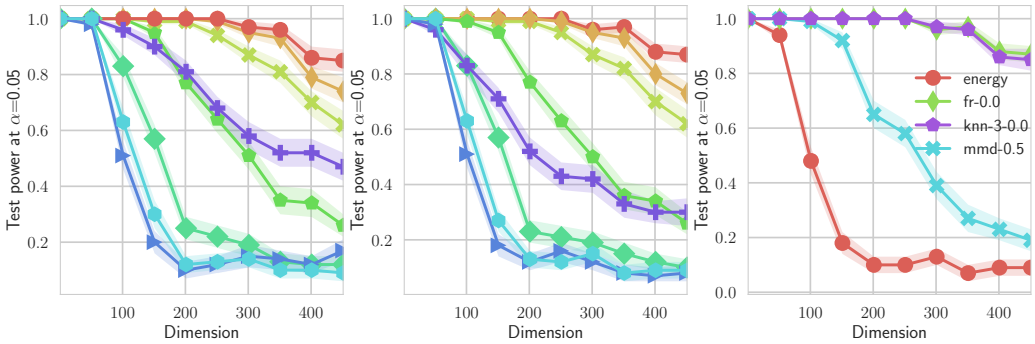
Model	Step size	Batch size	Epochs
Figure 1a	10^{-4}	256	500
Figure 1b	10^{-4}	256	500
Figure 1c	10^{-4}	256	500
Figure 2a	$10^{-3}, 10^{-4}$	512	500, 500
Figure 2b	$10^{-4}, 10^{-4}$	128	200, 200



(a) Power against the alternative $(\mu = 0.5, \sigma = 1)$ from $n_1 = n_2 = 128$ samples.

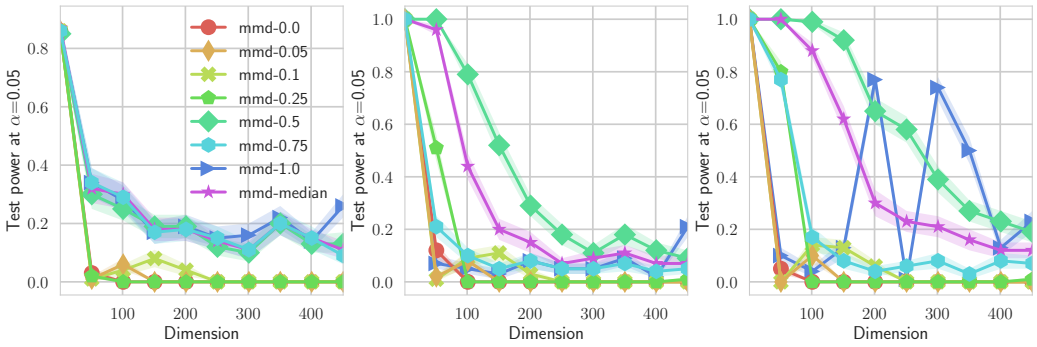


(b) Power against the alternative $(\mu = 0, \sigma = 3)$ from $n_1 = n_2 = 128$ samples.



(c) Power against the alternative $(\mu = 0, \sigma = 3)$ from $n_1 = n_2 = 256$ samples.

Figure 3: Test power when comparing two normal distributions. In the first two columns we present the 3-NN and FR tests as we vary λ — we use fr-γ for $\lambda = d^{\gamma}$, and fr-ct for the classical test (analogously for 3-NN). The legends presented in the first row are consistent across the respective columns. The last column compares the best performing of these tests with the best performing MMD tests (the remaining MMD plots are provided in Figure 4). Note that our smoothed tests have the largest power, and they significantly outperform their classical counterparts.



(a) $\mu = 0.5, \sigma = 1, n_1 = 128$. (b) $\mu = 0, \sigma = 3, n_1 = 128$. (c) $\mu = 0, \sigma = 3, n_1 = 256$.

Figure 4: The different MMD tests on the three setups in Figure 3. The legend is consistent across the panels.