# Telescoping Density-Ratio Estimation

Benjamin Rhodes  
School of Informatics  
University of Edinburgh  
ben.rhodes@ed.ac.uk

Kai Xu  
School of Informatics  
University of Edinburgh  
kai.xu@ed.ac.uk

Michael U. Gutmann  
School of Informatics  
University of Edinburgh  
michael.gutmann@ed.ac.uk

## Abstract

Density-ratio estimation via classification is a cornerstone of unsupervised learning. It has provided the foundation for state-of-the-art methods in representation learning and generative modelling, with the number of use-cases continuing to proliferate. However, it suffers from a critical limitation: it fails to accurately estimate ratios \( p/q \) for which the two densities differ significantly. Empirically, we find this occurs whenever the KL divergence between \( p \) and \( q \) exceeds tens of nats. To resolve this limitation, we introduce a new framework, telescoping density-ratio estimation (TRE), that enables the estimation of ratios between highly dissimilar densities in high-dimensional spaces. Our experiments demonstrate that TRE can yield substantial improvements over existing single-ratio methods for mutual information estimation, representation learning and energy-based modelling.

## 1 Introduction

Unsupervised learning via density-ratio estimation is a powerful paradigm in machine learning [60] that continues to be a source of major progress in the field. It consists of estimating the ratio \( p/q \) from their samples without separately estimating the numerator and denominator. A common way to achieve this is to train a neural network classifier to distinguish between the two sets of samples, since for many loss functions the ratio \( p/q \) can be extracted from the optimal classifier [60, 21, 41]. This discriminative approach has been leveraged in diverse areas such as covariate shift adaptation [59, 63], energy-based modelling [22, 4, 53, 64, 36, 19], generative adversarial networks [15, 47, 43], bias correction for generative models [20, 18], likelihood-free inference [50, 62, 8, 13], mutual-information estimation [2], representation learning [29, 30, 48, 25, 27], Bayesian experimental design [33, 34] and off-policy reward estimation in reinforcement learning [39]. Across this diverse set of applications, density-ratio based methods have consistently yielded state-of-the-art results.

Despite the successes of discriminative density-ratio estimation, many existing loss functions share a severe limitation. Whenever the ‘gap’ between \( p \) and \( q \) is large, the classifier can obtain almost perfect accuracy with a relatively poor estimate of the density ratio. We refer to this failure mode as the density-chasm problem—see Figure 1a for an illustration. We observe empirically that the density-chasm problem manifests whenever the KL divergence \( D_{KL}(p \parallel q) \) exceeds \( \sim 20 \) nats.\(^1\) This observation accords with recent findings in the mutual information literature regarding the limitations of density-ratio based estimators of the KL [40, 52, 57]. In high dimensions, it can easily occur that two densities \( p \) and \( q \) will have a KL-divergence measuring in the hundreds of nats, and so the ratio may be virtually intractable to estimate with existing techniques.

In this paper, we propose a new framework for estimating density-ratios that can overcome the density-chasm problem. Our solution uses a ‘divide-and-conquer’ strategy composed of two steps. The first step is to gradually transport samples from \( p \) to samples from \( q \), creating a chain of intermediate datasets. We then estimate the density-ratio between consecutive datasets along this

---

\(^1\)‘nat’ being a unit of information measured using the natural logarithm (base \( e \))
We show this for two important applications: representation learning via mutual information (MI).

We empirically demonstrate that TRE can accurately estimate density-ratios using deep neural networks on high-dimensional problems, significantly outperforming existing single-ratio methods.

Figure 1: Illustration of standard density-ratio estimation vs. telescoping density-ratio estimation.

We empirically demonstrate that TRE can accurately estimate density-ratios using deep neural networks on high-dimensional problems, significantly outperforming existing single-ratio methods. We show this for two important applications: representation learning via mutual information (MI) estimation and the learning of energy-based models (EBMs).

In the context of mutual information estimation, we show that TRE can accurately estimate large MI values of 30+ nats, which is recognised to be an outstanding problem in the literature. However, obtaining accurate MI estimates is often not our sole objective; we also care about learning representations from e.g. audio or image data that are useful for downstream tasks such as classification or clustering. To this end, our experimental results for representation learning confirm that TRE offers substantial gains over a range of existing single-ratio baselines.

In the context of energy-based modelling, we show that TRE can be viewed as an extension of noise-contrastive estimation that more efficiently scales to high-dimensional data. Whilst energy-based modelling has been a topic of interest in the machine learning community for some time, there has been a recent surge of interest, with a wave of new methods for learning deep EBMs in high dimensions. These methods have shown promising results for image and 3D shape synthesis, hybrid modelling, and modelling of exchangeable data.

(a) Density-ratio estimation between an extremely peaked Gaussian $p (\sigma = 10^{-6})$ and a broad Gaussian $q (\sigma = 1)$ using a single-parameter quadratic classifier (as detailed in section 4.1). Left: A log-log scale plot of the densities and their ratio. Note that $p(x)$ is not visible, since the ratio overlaps it. Right: the solid blue line is the finite-sample logistic loss (Eq. 2) for 10,000 samples. Despite the large sample size, the minimiser (dotted blue line) is far from optimal (dotted black line). The dotted red line is the newly introduced TRE solution, which almost perfectly overlaps with the dotted black line.

(b) Telescoping density-ratio estimation applied to the problem in (a), using the same 10,000 samples from $p$ and $q$. Top row: a collection of ratios, where $p_1$, $p_2$ and $p_3$ are constructed by deterministically interpolating between samples from $p$ and $q$. Bottom row: the logistic loss function for each ratio estimation problem. Observe that the finite-sample minimisers of each objective (red dotted lines) are either close to or exactly overlapping their optima (black dotted lines). After estimating each ratio, we then combine them by taking their product.
However, many of these methods result in expensive/challenging optimisation problems, since they rely on approximate Markov chain Monte Carlo (MCMC) sampling during learning [10,16,68], or on adversarial optimisation [6,17,68]. In contrast, TRE requires no MCMC during learning and uses a well-defined, non-adversarial, objective function. Moreover, as we show in our mutual information experiments, TRE is applicable to discrete data, whereas all other recent EBM methods only work for continuous random variables. Applicability to discrete data makes TRE especially promising for domains such as natural language processing, where noise-contrastive estimation has been widely used [42,35,1].

2 Discriminative ratio estimation and the density-chasm problem

Suppose $p$ and $q$ are two densities for which we have samples, and that $q(x) > 0$ whenever $p(x) > 0$. We can estimate the density-ratio $r(x) = p(x)/q(x)$ by training a classifier to distinguish samples from $p$ and $q$ [23,60,22]. There are many choices for the loss function of the classifier [60,51,21,41,52], but in this paper we concentrate on the widely used logistic loss

$$L(\theta) = -\mathbb{E}_{x_1 \sim p} \log \left( \frac{r(x_1; \theta)}{1 + r(x_1; \theta)} \right) - \mathbb{E}_{x_2 \sim q} \log \left( \frac{1}{1 + r(x_2; \theta)} \right),$$

(1)

where $r(x; \theta)$ is a non-negative ratio estimating model. To enforce non-negativity, $r$ is typically expressed as the exponential of an unconstrained function such as a neural network. For a correctly specified model, the minimiser of this loss, $\theta^*$, satisfies $r(x; \theta^*) = p(x)/q(x)$, without needing any normalisation constraints [22]. Other classification losses do not always have this self-normalising property, but only yield an estimate proportional to the true ratio—see e.g. [52].

The density-chasm problem

We experimentally find that density-ratio estimation via classification typically works well when $p$ and $q$ are ‘close’ e.g. the KL divergence between them is less than $\sim 20$ nats. However, for sufficiently large gaps, which we refer to as density-chasms, the ratio estimator is often severely inaccurate. This raises the obvious question: what is the cause of such inaccuracy?

There are many possible sources of error: the use of misspecified models, imperfect optimisation algorithms, and inaccuracy stemming from Monte Carlo approximations of the expectations in (1). We argue that this mundane final point—Monte Carlo error due to finite sample size—is actually sufficient for inducing the density-chasm problem. Figure [14] depicts a toy problem for which the model is well-specified, and because it is 1-dimensional (w.r.t. $\theta$), optimisation is straightforward using grid-search. And yet, if we use a sample size of $n = 10,000$ and minimise the finite-sample loss

$$\mathcal{L}^n(\theta) = \sum_{i=1}^{n} - \log \left( \frac{r(x_i^1; \theta)}{1 + r(x_i^1; \theta)} \right) - \log \left( \frac{1}{1 + r(x_i^2; \theta)} \right),$$

(2)

we obtain an estimate $\hat{\theta}$ that is far from the asymptotic minimiser $\theta^* = \arg \min \mathcal{L}(\theta)$. Repeating this same experiment for different sample sizes, we can empirically measure the method’s sample efficiency, which is plotted as the blue curve in Figure 2. For the regime plotted, we see that an exponential increase in sample size only yields a linear decrease in estimation error. This empirical result is concordant with theoretical findings that density-ratio based lower bounds on KL divergences are only tight for sample sizes exponential in the the number of nats [40].

Whilst we focus on the logistic loss, we believe the density chasm problem is a broader phenomenon. As shown in the appendix, the issues identified in Figure 1 and the sample inefficiency seen in Figure 2 also occur for other commonly used discriminative loss functions.

Thus, when faced with the density-chasm problem, simply increasing the sample size is a highly inefficient solution and not always possible in practice. This begs the question: is there a more intelligent way of using a fixed set of samples from $p$ and $q$ to estimate the ratio?
3 Telescoping density-ratio estimation

We introduce a new framework for estimating density-ratios \( p/q \) that can overcome the density-chasm problem in a sample-efficient manner. Intuitively, the density-chasm problem arises whenever classifying between \( p \) and \( q \) is ‘too easy’. This suggests that it may be fruitful to decompose the task into a collection of harder sub-tasks.

For convenience, we make the notational switch \( p \equiv p_0, \ q \equiv p_m \) (which we will keep going forward), and expand the ratio via a telescoping product

\[
\frac{p_0(x)}{p_m(x)} = \frac{p_0(x)}{p_1(x)} \frac{p_1(x)}{p_2(x)} \cdots \frac{p_{m-1}(x)}{p_m(x)} = r(x; \theta),
\]

where, ideally, each \( p_k \) is chosen such that a classifier cannot easily distinguish it from its two neighbouring densities. Instead of attempting to build one large ‘bridge’ (i.e. density-ratio) across the density-chasm, we propose to build many small bridges between intermediate ‘waymark’ distributions. The two key components of the method are therefore:

1. **Waymark creation.** We require a method for **gradually** transporting samples \( \{x_1^0, \ldots, x_m^0\} \) from \( p_0 \) to samples \( \{x_1^m, \ldots, x_m^m\} \) from \( p_m \). At each step in the transportation, we obtain a new dataset \( \{x_1^k, \ldots, x_m^k\} \) where \( k \in \{0, \ldots, m\} \). Each intermediate dataset can be thought of as samples from an implicit distribution \( p_k \), which we refer to as a waymark distribution.

2. **Bridge-building:** A method for learning a set of parametrised density-ratios between consecutive pairs of waymarks \( \frac{p_k(x; \theta_k)}{p_{k+1}(x)} \approx \frac{p_k(x)}{p_{k+1}(x)} \) for \( k = 0, \ldots, m-1 \), where each bridge \( r_k \) is a non-negative function. We refer to these ratio estimating models as *bridges*. Note that the parameters of the bridges, \( \{\theta_k\}_{k=0}^{m-1} \), can be totally independent or they can be partially shared.

An estimate of the original ratio is then given by the product of the bridges

\[
r(x; \theta) = \prod_{k=0}^{m-1} r_k(x; \theta_k) \approx \prod_{k=0}^{m-1} \frac{p_k(x)}{p_{k+1}(x)} = \frac{p_0(x)}{p_m(x)},
\]

where \( \theta \) is the concatenation of all \( \theta_k \) vectors. Because of the telescoping product in (4), we refer to the method as Telescoping density-Ratio Estimation (TRE).

TRE has conceptual ties with a range of methods in optimisation, statistical physics and machine learning that leverage sequences of intermediate distributions, typically between a complex density \( p \) and a simple tractable density \( q \). Of particular note are the methods of Simulated Annealing [32], Bridge Sampling & Path Sampling [19] and Annealed Importance Sampling (AIS) [45]. Whilst none of these methods estimate density ratios, and thus serve fundamentally different purposes, they leverage similar ideas. In particular, AIS also computes a chain of density-ratios between artificially constructed intermediate distributions. It typically does this by first defining explicit expressions for the intermediate densities, and then trying to obtain samples via MCMC. In contrast, TRE *implicitly* defines the intermediate distributions via samples and then tries to learn the ratios. Additionally, in TRE we would like to evaluate the learned ratios in (4) at the same input \( x \) while AIS should only evaluate a ratio \( r_k \) at ‘local’ samples from e.g. \( p_k \).

### 3.1 Waymark creation

In this paper, we consider two simple, deterministic waymark creation mechanisms: **linear combinations** and **dimension-wise mixing**. We find these mechanisms yield good performance and are computationally cheap. However, we note that other mechanisms are possible, and are a promising topic for future work.

**Linear combinations.** Given a random pair \( x_0 \sim p_0 \) and \( x_m \sim p_m \), define the \( k \)th waymark via

\[
x_k = \sqrt{1 - \alpha_k^2} x_0 + \alpha_k x_m, \quad k = 0, \ldots, m
\]

where the \( \alpha_k \) form an increasing sequence from 0 to 1, which control the distance of \( x_k \) from \( x_0 \). For all of our experiments (except, for illustration purposes, those depicted in Figure[1]), each dimension of
$p_0$ and $p_m$ has the same variance and the coefficients in (5) are chosen to preserve this variance, with the goal being to match basic properties of the waymarks and thereby make consecutive classification problems harder.

**Dimension-wise mixing.** An alternative way to ‘mix’ two vectors is to concatenate different subsets of their dimensions. Given a $d$-length vector $x$, we can partition it into $m$ sub-vectors of length $d/m$, assuming $d$ is divisible by $m$. We denote this as $x = (x[1], \ldots, x[m])$, where each $x[i]$ has length $d/m$. Using this notation, define the $k$th waymark via

$$x_k = (x_m[1], \ldots, x_m[k], x_0[k+1], \ldots, x_0[m]) \quad k = 0, \ldots, m$$

(6)

where, again, $x_0 \sim p_0$ and $x_m \sim p_m$ are randomly paired.

**Number and spacing.** Given these two waymark generation mechanisms, we still need to decide the number of waymarks, $m$, and, in the case of linear combinations, how the $\alpha_k$ are spaced in the unit interval. We treat these quantities as hyperparameters, and demonstrate in the experiments (Section 4) that tuning them is feasible with a limited search budget.

### 3.2 Bridge-building

Each bridge $r_k(x; \theta_k)$ in (4) can be learned via binary classification using a logistic loss function as described in Section 2. Solving this collection of classification tasks is therefore a multi-task learning (MTL) problem—see [5] for a review. Two key questions in MTL are how to share parameters and how to define a joint objective function.

**Parameter sharing.** We break the construction of the bridges $r_k(x; \theta_k)$ into two stages: a (mostly) shared body computing hidden vectors $f_k(x)$ followed by bridge-specific heads. The body $f_k$ is a deep neural network with shared parameters and pre-activation per-hidden-unit scales and biases for each bridge (see appendix for details). Similar parameter sharing schemes have been successfully used in the multi-task learning literature [7, 11]. The heads map the hidden vectors $f_k(x)$ to the scalar $\log r_k(x; \theta_k)$. We use either linear or quadratic mappings depending on the application; the precise parameterisation is stated in each experiment section.

**TRE loss function.** The TRE loss function is given by the average of the $m$ logistic losses

$$\mathcal{L}_{\text{TRE}}(\theta) = \frac{1}{m} \sum_{k=0}^{m-1} \mathcal{L}_k(\theta_k),$$

(7)

$$\mathcal{L}_k(\theta_k) = -\mathbb{E}_{x_k \sim p_k} \log \left( \frac{r_k(x_k; \theta_k)}{1 + r_k(x_k; \theta_k)} \right) - \mathbb{E}_{x_{k+1} \sim p_{k+1}} \log \left( \frac{1}{1 + r_k(x_{k+1}; \theta_k)} \right).$$

(8)

This simple unweighted average works well empirically. More sophisticated multi-task weighting schemes exist [5], but preliminary experiments suggested they were not worth the extra complexity.

An important aspect of this loss function is that each ratio estimator $r_k$ sees different samples during training. In particular, $r_0$ sees samples close to the real data i.e. from $p_0$ and $p_1$, while the final ratio $r_{m-1}$ sees data from $p_{m-1}$ and $p_m$. This creates a potential mismatch between training and deployment, since after learning, we would like to evaluate all ratios at the same input $x$. In our experiments, we do not find this mismatch to be a problem, suggesting that each ratio, despite seeing different inputs during training, is able to generalise to new test points. We speculate that this generalisation is encouraged by parameter sharing, which allows each ratio-estimator to be indirectly influenced by samples from all waymark distributions. Nevertheless, we think a deeper analysis of this issue of generalisation deserves further work.

### 3.3 TRE applied to mutual information estimation

The mutual information (MI) between two random variables $u$ and $v$ can be written as

$$I(u, v) = \mathbb{E}_{p(u, v)} \left[ \log r(u, v) \right], \quad r(u, v) = \frac{p(u, v)}{p(u)p(v)}.$$  

(9)

\[\text{Footnote 5: For MI estimation this always holds, for energy-based modelling this is enforceable via the choice of } p_m.\]

\[\text{Footnote 6: For simplicity, we suppress the parameters of } f_k, \text{ and will do the same for } r_k \text{ in the experiments section.}\]
Given samples from the joint density \( p(u, v) \), one obtains samples from the product-of-marginals \( p(u)p(v) \) by shuffling the \( v \) vectors across the dataset. This then enables standard density-ratio estimation to be performed.

For TRE, we require waymark samples. To generate these, we take a sample from the joint, \( x_0 = (u, v_0) \), and a sample from the product-of-marginals, \( x_m = (u, v_m) \), where \( u \) is held fixed and only \( v \) is altered. We then apply a waymark construction mechanism from Section 3.1 to generate \( x_k = (u, v_k) \), for \( k = 0, \ldots, m \).

### 3.4 TRE applied to energy-based modelling

An energy-based model (EBM) is a flexible parametric family \( \{ \phi(x; \theta) \} \) of non-negative functions, where each function is proportional to a probability-density. Given samples from a data distribution \( p(x) \), the goal of energy-based modelling is to find a parameter \( \theta^* \) such that \( \phi(x; \theta^*) \) is ‘close’ to \( c p(x) \), for some positive constant \( c \).

In this paper, we consider EBMs of the form \( \phi(x; \theta) = r(x; \theta)q(x) \), where \( q \) is a known density (e.g. a Gaussian or normalising flow) that we can sample from, and \( r \) is an unconstrained positive function. Given this parameterisation, the optimal \( r \) simply equals the density-ratio \( p(x)/q(x) \), and hence the problem of learning an EBM becomes the problem of estimating a density-ratio, which can be solved via TRE. We note that, since TRE actually estimates a product of ratios as stated in Equation 4, the final EBM will be a product-of-experts model \( \{26\} \) of the form \( \phi(x; \theta) = \prod_{k=0}^{m-1} r_k(x; \theta_k)q(x) \).

The estimation of EBMs via density-ratio estimation has been studied in multiple prior works, including noise-contrastive estimation (NCE) \( \{22\} \), which has many appealing theoretical properties \( \{22, 54, 65\} \). Following NCE, we will refer to the known density \( q \) as the ‘noise distribution’.

### 4 Experiments

We include two toy examples illustrating both the correctness of TRE and the fact that it can solve problems which verge on the intractable for standard density ratio estimation. We then demonstrate the utility of TRE on two high-dimensional complex tasks, providing clear evidence that it substantially improves on standard single-ratio baselines.

For experiments with continuous random variables, we use the linear combination waymark mechanisms in \( \{5\} \); otherwise, for discrete variables, we use dimension-wise mixing \( \{6\} \). For the linear combination mechanism, we collapse the \( \alpha_k \) into a single spacing hyperparameter, and grid-search over this value, along with the number of waymarks. Details are in the appendix.

#### 4.1 1d peaked ratio

The basic setup is stated in Figure 1a. For TRE, we use quadratic bridges of the form \( \log r_k(x) = w_k x^2 + b_k \), where \( b_k \) is set to its ground truth value (as derived in appendix), and \( w_k \) is reparametrised as \( \exp(\theta_k) \) to avoid unnecessary log-scales in Figure 1. The single ratio-estimation results use the same parameterisation (dropping the subscript \( k \)). Figure 2 shows the full results. These sample efficiency curves clearly demonstrate that, across all sample sizes, TRE is significantly more accurate than single ratio estimation. In fact, TRE obtains a better solution with 100 samples than single-ratio estimation does with 100,000 samples: a three orders of magnitude improvement.

#### 4.2 High-dimensional ratio with large MI

This toy problem has been widely used in the mutual information literature \( \{2, 52\} \). Let \( x \in \mathbb{R}^{2d} \) be a Gaussian random variable, with block-diagonal covariance matrix, where each block is \( 2 \times 2 \) with 1 on the diagonal and 0.8 on the off-diagonal. We then estimate the ratio between this Gaussian and a standard normal

Figure 3: High-dimensional Gaussian results, showing estimated MI as a function of the dimensionality. Errors bars were computed over 5 random seeds, but are too small to see.
distribution. This problem can be viewed as an MI estimation task or an energy-based modelling task—see the appendix for full details.

We apply TRE using quadratic bridges of the form: \( \log r_k(x) = x^T W_k x + b_k \). The results in Figure 3 show that single ratio estimation becomes severely inaccurate for MI values greater than 20 nats. In contrast, TRE can accurately estimate MI values as large as 80 nats for 320 dimensional variables. To our knowledge, TRE is the first discriminative MI estimation method that can scale this gracefully.

### 4.3 MI estimation & representation learning on SpatialMultiOmniglot

We applied TRE to the SpatialMultiOmniglot problem taken from [49] where characters from Omniglot are spatially stacked in an \( n \times n \) grid, where each grid position contains characters from a fixed alphabet. Following [49], the individual pixel values of the characters are not considered random variables; rather, we treat the grid as a collection of \( n^2 \) categorical random variables whose realisations are the characters from the respective alphabet. Pairs of grids, \((u, v)\), are then formed such that corresponding grid-positions contain alphabetically consecutive characters. Given this setup, the ground truth MI can be calculated (see appendix).

Each bridge in TRE uses a separable architecture [52] given by \( \log r_k(u, v) = g(u)^T W_k f_k(v) \), where \( g \) and \( f_k \) are 14-layer convolutional ResNets [24] and \( f_k \) uses the parameter-sharing scheme described in Section 3.2. We note that separable architectures are standard in the MI-based representation learning literature [52]. We construct waymarks using the dimension-wise mixing mechanism (6) with \( m = n^2 \) (i.e. one dimension is mixed at a time).

After learning, we adopt a standard linear evaluation protocol (see e.g. [48]), where we train supervised linear classifiers on top of the output layer \( g(u) \) to predict the alphabetic position of each character in \( u \). We compare our results to those reported in [49]. Specifically, we report their baseline method—contrastive predictive coding (CPC) [48], a state-of-the-art representation learning method based on single density-ratio estimation—along with their variant, Wasserstein predictive coding (WPC).

Figure 4 shows the results. The left plot shows that only TRE can accurately estimate high MI values of \( \sim 35 \) nats. The representation learning results (right) show that all single density-ratio baselines degrade significantly in performance as we increase the number of characters in a grid (and hence increase the MI). In contrast, TRE always obtains greater than 97% accuracy.

### 4.4 Energy-based modelling on MNIST

As explained in Section 3.4, TRE can be used estimate an energy-based model of the form \( \phi(x; \theta) = \prod_{k=0}^{m-1} r_k(x; \theta_k) q(x) \), where \( q \) is a pre-specified ‘noise’ distribution from which we can sample, and the product of ratios is given by TRE. In this section, we demonstrate that such an approach can

---

3We mirror their experimental setup as accurately as possible, however we were unable to obtain their code.

4[49] do not provide MI estimates for CPC & WPC, but [52] shows that they are bounded by log batch-size.
Table 1: Average negative log-likelihood in bits per dimension (bpd, smaller is better). Exact computation is intractable for EBMs, but we provide 3 estimates: Direct/RAISE/AIS. The ‘Direct’ estimate uses the NCE/TRE approximate normalising constant.

<table>
<thead>
<tr>
<th>Noise distribution</th>
<th>Noise</th>
<th>Single ratio (NCE)</th>
<th>TRE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Direct</td>
<td>RAISE</td>
<td>AIS</td>
</tr>
<tr>
<td>Gaussian</td>
<td>2.01</td>
<td>1.96</td>
<td>1.99</td>
</tr>
<tr>
<td>Gaussian Copula</td>
<td>1.40</td>
<td>1.33</td>
<td>1.48</td>
</tr>
<tr>
<td>RQ-NSF</td>
<td>1.12</td>
<td>1.09</td>
<td>1.10</td>
</tr>
</tbody>
</table>

Figure 5: MNIST samples. Each row pertains to a particular noise distribution. The first block shows exact samples from that distribution. The second & third blocks show MCMC samples from an EBM learned with NCE & TRE, respectively.

We scale to high-dimensional data, by learning energy-based models of the MNIST handwritten digit dataset [37]. We consider three choices of the noise distribution: a multivariate Gaussian, a Gaussian copula and a rational-quadratic neural spline flow (RQ-NSF) [12] with coupling layers [9, 31]. Each distribution is first fitted to the data via maximum likelihood estimation—see appendix for details.

Each of these noise distributions can be expressed as an invertible transformation of a standard normal distribution. That is, each random variable has the form \( F(z) \), where \( z \sim N(0, I) \). Since \( F \) already encodes useful information about the data distribution, it makes sense to leverage this when constructing the waymarks in TRE. Specifically, we can generate linear combination waymarks via \( x_k = \sqrt{1 - \alpha_k^2} F^{-1}(x_0) + \alpha_k F^{-1}(x_m) \).

For a Gaussian, \( F \) is linear, and hence (10) is identical to the original waymark mechanism in (5).

We use the parameter sharing scheme from Section 3.2 together with quadratic heads. This gives \( \log r_k(x) = -f_k(x)^T W_k f_k(x) - f_k(x)^T b_k - c_k \), where we set \( f_k \) to be an 18-layer convolutional Resnet and constrain \( W_k \) to be positive definite. This constraint enforces an upper limit on the log-density of the EBM, which has been useful in other work [44, 46], and improves results here.

We evaluate the learned EBMs quantitatively via estimated log-likelihood in Table 1 and qualitatively via random samples from the model in Figure 5. For both of these evaluations, we employ NUTS [28] to perform annealed MCMC sampling as explained in the appendix. This annealing procedure provides two estimators of the log-likelihood: the Annealed Importance Sampling (AIS) estimator [45] and the more conservative Reverse Annealed Importance Sampling Estimator (RAISE) [3].

The results in Table 1 and Figure 5 show that single ratio estimation performs poorly in high-dimensions for simple choices of the noise distribution, and only works well if we use a complex neural density-estimator (RQ-NSF). This illustrates the density-chasm problem explained in Section 2. In contrast, TRE yields improvements for all choices of the noise, as measured by the approximate log-likelihood and the visual fidelity of the samples. TRE’s improvement over the Gaussian noise distribution is particularly large: the bits per dimension (bpd) is around 0.66 lower, corresponding to an improvement of roughly 360 nats. Moreover, the samples are significantly more coherent, and appear to be of higher fidelity than the RQ-NSF samples despite the fact that TRE (with Gaussian noise) has a worse log-likelihood. This final point is not contradictory since log-likelihood and sample quality are known to be only loosely connected [61].

---

6We emphasise here that the quality of the RQ-NSF model depends on the exact architecture. A larger model may yield better samples. Thus, we do not claim that TRE generally yields superior results in any sense.
Finally, we analysed the sensitivity of our results to the construction of the waymarks and include the results in the appendix. Using TRE with a copula noise distribution as an illustrative case, we found that varying the number of waymarks between 5-30 caused only minor changes in the approximate log-likelihoods, no greater than 0.03 bpd. We also found that if we omit the z-space waymark mechanism in (10), and work in x-space, then TRE’s negative log-likelihood increases to 1.33 bpd, as measured by RAISE. This is still significantly better than single-ratio estimation, but does show that the quality of the results depends on the exact waymark mechanism.

5 Conclusion

We introduced a new framework—Telescoping density-Ratio Estimation (TRE)—for learning density-ratios that, unlike existing discriminative methods, can accurately estimate ratios between extremely different densities in high-dimensions.

TRE admits many exciting directions for future work. Firstly, we would like a deeper theoretical understanding of why it is so much more sample-efficient than standard density-ratio estimation. The relationship between TRE and standard methods is structurally similar to the relationship between annealed importance sampling and standard importance sampling. Thus, exploring this connection further may be fruitful. Relatedly, we believe that TRE would benefit from further research on waymark mechanisms. We presented simple mechanisms that have clear utility for both discrete and continuous-valued data. However, we suspect more sophisticated choices may yield improvements, especially if one can leverage domain or task-specific assumptions to intelligently decompose the density-ratio problem. Lastly, whilst this paper has focused on the logistic loss, it would be interesting to more deeply investigate TRE with other discriminative loss functions.

Broader Impact

As outlined in the introduction, density-ratio estimation is a foundational tool in machine learning with diverse applications. Our work, which improves density-ratio estimation, may therefore increase the scope and power of a wide spectrum of techniques used both in research and real-world settings. The broad utility of our contribution makes it challenging to concretely assess the societal impact of the work. However, we do discuss here two applications of density-ratio estimation with obvious potential for positive & negative impacts on society.

Generative Adversarial Networks [15] are a popular class of models which are often trained via density-ratio estimation and are able to generate photo-realistic image/video content. To the extent that TRE can enhance GAN training (a topic we do not treat in this paper), our work could conceivably lead to enhanced ‘deepfakes’, which can be maliciously used in fake-news or identity fraud.

More positively, density-ratio estimation is being used to correct for dataset bias, including the presence of skewed demographic factors like race and gender [18]. While we are excited about such applications, we emphasise that density-ratio based methods are not a panacea; it is entirely possible for the technique to introduce new biases when correcting for existing ones. Future work should continue to be mindful of such a possibility, and look for ways to address the issue if it arises.

Acknowledgments and Disclosure of Funding

Benjamin Rhodes was supported in part by the EPSRC Centre for Doctoral Training in Data Science, funded by the UK Engineering and Physical Sciences Research Council (grant EP/L016427/1) and the University of Edinburgh. Kai was supported by Edinburgh Huawei Research Lab in the University of Edinburgh, funded by Huawei Technologies Co. Ltd.

References


Telescoping Density-Ratio Estimation: Supplementary Material

Benjamin Rhodes
School of Informatics
University of Edinburgh
ben.rhodes@ed.ac.uk

Kai Xu
School of Informatics
University of Edinburgh
kai.xu@ed.ac.uk

Michael U. Gutmann
School of Informatics
University of Edinburgh
michael.gutmann@ed.ac.uk

1 ResNet architectures with parameter sharing

<table>
<thead>
<tr>
<th>5 × 5 conv with 3 × 3 strides, 32n</th>
<th>3 × 3 conv, 64</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConditionalScaleShift</td>
<td>CondResBlock down, 64</td>
</tr>
<tr>
<td>CondResBlock down, 32n</td>
<td>AttentionBlock</td>
</tr>
<tr>
<td>CondResBlock, 32n</td>
<td>CondResBlock, 64</td>
</tr>
<tr>
<td>CondResBlock down, 64n</td>
<td>CondResBlock, 64</td>
</tr>
<tr>
<td>CondResBlock, 64n</td>
<td>CondResBlock down, 128</td>
</tr>
<tr>
<td>CondResBlock down, 64n</td>
<td>CondResBlock, 128</td>
</tr>
<tr>
<td>CondResBlock, 64n</td>
<td>CondResBlock down, 128</td>
</tr>
<tr>
<td>CondResBlock, 128</td>
<td>GlobalSumPooling</td>
</tr>
<tr>
<td></td>
<td>Dense, 300n</td>
</tr>
</tbody>
</table>

(a) SpatialMultiOmniglot architecture. The multiplier n refers to the width/height of a datapoint, which is an n × n grid.

<table>
<thead>
<tr>
<th>3 × 3 conv, 64</th>
</tr>
</thead>
<tbody>
<tr>
<td>ConditionalScaleShift</td>
</tr>
<tr>
<td>CondResBlock down, 64</td>
</tr>
<tr>
<td>CondResBlock, 64</td>
</tr>
<tr>
<td>CondResBlock down, 64</td>
</tr>
<tr>
<td>CondResBlock, 64</td>
</tr>
<tr>
<td>CondResBlock down, 64</td>
</tr>
<tr>
<td>CondResBlock, 64</td>
</tr>
<tr>
<td>CondResBlock down, 128</td>
</tr>
<tr>
<td>CondResBlock, 128</td>
</tr>
<tr>
<td>CondResBlock down, 128</td>
</tr>
<tr>
<td>CondResBlock, 128</td>
</tr>
<tr>
<td>GlobalSumPooling</td>
</tr>
<tr>
<td>Dense, 128</td>
</tr>
</tbody>
</table>

(b) MNIST architecture.

(c) CondResBlock. Dashed boxes denote layers that are not always present. The 1 × 1 conv layer (and the associated CondScaleShift) is only used in blocks where the channel size is altered. The 2 × 2 pool layer is only used for ‘down’ blocks.

Figure 1: Convolutional ResNet architectures

In Figure 1, we give the exact architectures for the \( f_k \) used in the two high-dimensional experiments on SpatialMultiOmniglot and MNIST. These \( f_k \) output a hidden vector for the \( k^{th} \) bridge, which is then mapped to the scalar value of the log-ratio, as stated in each experiment section. All convolution operations share their parameters across the bridges, and are thus independent of \( k \).

The only difference between our conditional residual blocks (i.e. ‘CondResBlocks’) and a standard residual block is the use of ‘ConditionalScaleShift’ layers. These layers map a hidden vector \( z_k \) to a hidden vector of the same size, \( z'_k \), via

\[
z'_k = s_k \odot z_k + b_k
\]

where \( s_k \) and \( b_k \) are bridge-specific parameters and \( \odot \) denotes element-wise multiplication. This operation could be thought of as class-conditional Batch Normalisation (BN) without the normalisation. We did not investigate the use of BN, since many energy-based modelling papers (e.g. [4])

found it to harm performance. We did perform preliminary experiments with Instance Normalisation [20] in the context of energy-based modelling, finding it to be harmful to performance.

For the MNIST energy-based modelling experiments, we use average pooling operations since other work [19, 4] has found this to produce higher quality samples than max pooling. For the SpatialMultiOmniglot experiments, we grid-search over average pooling and max pooling. For both sets of experiments, we use LeakyRelu activations with a slope of 0.3.

The MNIST architecture includes an attention block [21] which has been used in GANs to model long-range dependencies in the input image. We found that this attention layer did not yield improvements in estimated log-likelihood, but we think it may yield slightly more globally coherent samples. We note that that another commonly used feature in recent GAN and EBM architectures is Spectral Normalisation (SN) [13]. Our preliminary experiments suggested that SN was not beneficial for performance. That said, all of our negative results should be taken with a grain of salt, given the preliminary nature of the experiments.

2 Waymark number and spacing

As stated in the main text, the number and (in the case of linear combinations) the spacing of the waymarks are treated as hyperparameters. Finding good values of these hyperparameters is made simpler by the following observations.

- If any of the TRE logistic losses saturate close to 0 during learning, then this indicates that the density-chasm problem has occurred for that bridge, and we can terminate the run.
- As illustrated by our sensitivity analysis for MNIST (see Figure 5) it seems that, past a certain point, performance plateaus with the addition of extra waymarks. The fact that it plateaus, and does not decrease, is good news since it indicates that there is little risk of ‘overshooting’, and obtaining a bad model by having too many waymarks.

We now recall the linear combinations waymark mechanism, given by

\[ x_k = \sqrt{1 - \alpha_k^2} x_0 + \alpha_k x_m, \quad k = 0, \ldots, m. \]  

(2)

where \( m \) is the number of waymarks. We consider two ways of reducing the coefficients \( \alpha_k \) to a function of a single spacing hyperparameter \( p \) via

\[ \alpha_k = \left( \frac{k}{m} \right)^p, \quad k = 0, \ldots, m, \]  

(3)

\[ \alpha_k = \begin{cases} \left( \frac{k}{m} \right)^p, & \text{for } k \leq m/2 \\ 1 - \left( \frac{m - k}{m} \right)^p, & \text{for } k \geq m/2 \end{cases} \quad k = 0, \ldots, m. \]  

(4)

Both mechanisms yield linearly spaced \( \alpha_k \) when \( p = 1 \). For the first mechanism in (3), setting \( p > 1 \) means the gaps between waymarks increase with \( k \) (and conversely decrease if \( p < 1 \)). The spacing mechanism in (4) is a kind of symmetrised version of (3).

Table 1 shows the grid-searches we performed for all experiments. We note that these weren’t always all performed in parallel. When using linear combinations, we typically set \( p = 1 \) initially and searched over values of \( m \). If, for all values of \( m \) tested, one of the TRE logistic losses saturated close to 0, then we would expand our search space and test different values of \( p \).

<table>
<thead>
<tr>
<th>experiment</th>
<th>mechanism</th>
<th>( m )</th>
<th>spacing</th>
<th>( p )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1d peaked ratio</td>
<td>linear combo</td>
<td>4</td>
<td>{1, 2, \ldots, 7, 8}</td>
<td></td>
</tr>
<tr>
<td>high dim, high MI</td>
<td>linear combo</td>
<td>( \frac{d}{9} \times {1, 2, 3, 4} )</td>
<td>( d )</td>
<td>{1}</td>
</tr>
<tr>
<td>SpatialMultiOmniglot</td>
<td>dim-wise mix</td>
<td>( d )</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td>MNIST (z-space)</td>
<td>linear combo</td>
<td>{5, 10, 15, 20, 25, 30}</td>
<td>{5}</td>
<td>{1}</td>
</tr>
<tr>
<td>MNIST (x-space)</td>
<td>linear combo</td>
<td>{5, 10, 15, 20, 25, 30}</td>
<td>{3, 4}</td>
<td>{1, 1.5, 2}</td>
</tr>
</tbody>
</table>

Note: \( d \) refers to the dimensionality of the dataset, which is varied for certain experiments.
3 Minibatching

Recall that the TRE loss is a sum of logistic losses:

\[
\mathcal{L}_{\text{TRE}}(\theta) = \frac{1}{m} \sum_{k=0}^{m-1} \mathcal{L}_k(\theta_k),
\]

(5)

\[
\mathcal{L}_k(\theta_k) = -E_{x_k \sim p_k} \log \left( \frac{r_k(x_k; \theta_k)}{1 + r_k(x_k; \theta_k)} \right) - E_{x_{k+1} \sim p_{k+1}} \log \left( \frac{1}{1 + r_k(x_{k+1}; \theta_k)} \right).
\]

(6)

When generating minibatch estimates of this loss, we can either sample from each \(p_k\) independently, or we can couple the samples. By ‘couple’, we mean first drawing \(B\) samples each from \(p_0\) and \(p_m\), randomly pairing members from each set, and then, for each pair, constructing all possible intermediate waymark samples to obtain a final minibatch of size \(B \times M\). Coupling in this way means that the gradient of (5) w.r.t. \(\theta\) is estimated using shared sources of randomness, which can act as a form of variance reduction [14].

In all of our experiments, we use coupling when forming minibatches, since we found it to be useful in some preliminary investigations. However, coupling does have memory costs: the number of independent samples drawn from the data distribution, \(B\), may need to be very small for the full minibatch, \(B \times M\), to fit into memory. We speculate that as \(B\) becomes sufficiently small, coupled minibatches will produce inferior results to non-coupled minibatches (which can use a greater number of independent real data samples). Empirical investigation of this claim is left to future work.

4 1d peaked ratio toy experiment

In this experiment we estimate the ratio \(p_0/p_m\), where both densities are Gaussian, \(p_0 = \mathcal{N}(0, \sigma_0^2)\) and \(p_m = \mathcal{N}(0, \sigma_m^2)\), where \(\sigma_0 = 10^{-6}\) and \(\sigma_m = 1\). We generate waymarks using the linear combinations mechanism [2], which implies that each waymark distribution is Gaussian, since linear combinations of Gaussian random variables are also Gaussian. Specifically, the waymark distributions have the form

\[
p_k(x) = \mathcal{N}(x; 0, \sigma_k^2), \quad \text{where } \sigma_k = \left[ (1 - \alpha_k^2) \sigma_0^2 + \alpha_k^2 \sigma_m^2 \right]^{\frac{1}{2}}.
\]

(7)

where the \(\sigma_k\) form an increasing sequence between \(\sigma_0\) and \(\sigma_m\). The log-ratio between two waymark distributions is therefore given by

\[
\log \frac{p_k(x)}{p_{k+1}(x)} = \log \left( \frac{\sigma_{k+1}}{\sigma_k} \right) + \left( \frac{1}{2 \sigma_{k+1}^2} - \frac{1}{2 \sigma_k^2} \right) x^2.
\]

(8)

We parameterise the bridges in TRE as

\[
\log r_k(x; \theta_k) = \log \left( \frac{\sigma_{k+1}}{\sigma_k} \right) - \exp(\theta_k)x^2;
\]

(9)

where the quadratic coefficient \(\exp(\theta_k)\) is always negative. We note that this model is well-specified since it contains the ground-truth solution in (8).

The bridges can then be combined via summation to provide an estimate of the original log-ratio

\[
\log \frac{p_0(x)}{p_m(x)} \approx \sum_{k=0}^{m-1} \log r_k(x; \theta_k)
\]

(10)

\[
= \log \left( \frac{\sigma_m}{\sigma_0} \right) - \sum_{k=0}^{m-1} \exp(\theta_k)x^2
\]

(11)

\[
= \log \left( \frac{\sigma_m}{\sigma_0} \right) - \exp(\theta_{\text{TRE}})x^2
\]

(12)

Where \(\theta_{\text{TRE}} = \log(\sum_{k=0}^{m-1} \exp(\theta_k))\). We observe that (12) has the same form as (9) if we were to set \(m = 1\) in (9) (i.e. if we use a single bridge). Hence \(\theta_{\text{TRE}}\) can be directly compared to the parameter value we would obtain if we used single density-ratio estimation. This is precisely the comparison we make in Figure 1a and Figure 2 of the main text.
4.1 The density chasm problem for non-logistic loss functions

In the main paper, we illustrated the density-chasm problem for the logistic loss using the 1d peaked ratio experiment. Here, we illustrate precisely the same phenomenon for the NWJ/MINE-f loss [10, 11] and a Least Squares (LSQ) loss used by [12]. The loss functions are given by

\[
\mathcal{L}_{\text{NWJ}}(\theta) = -E_p \left[ \log r(x; \theta) \right] - 1 + E_q \left[ r(x; \theta) \right]
\]

(13)

\[
\mathcal{L}_{\text{LSQ}}(\theta) = \frac{1}{2} E_p \left[ \sigma(\log(r(x; \theta))) - 1 \right]^2 + \frac{1}{2} E_q \left[ \sigma(\log(r(x; \theta))) \right]^2
\]

(14)

where the \( \sigma \) in (14) denotes the sigmoid function.

In Figures 2 & 3 we can see how single-density ratio estimation performs when using the NWJ and LSQ loss functions for 10,000 samples. The loss curves display the same ‘saturation’ effect seen for the logistic loss, where many settings of the parameter yield an almost identical value of the loss. Moreover, the minimiser of these saturated objectives is far from the ‘true’ minimiser (black dotted lines).

Figures 2 & 3 also show the performance of TRE when each bridge is estimated using the NWJ/LSQ losses. Each TRE loss has a quadratic bowl shape, where the finite-sample minimisers almost perfectly overlap with the true minimisers.

Finally, we plot sample efficiency curves for both the NWJ and LSQ losses, showing the results in Figure 4. We see that single density-ratio estimation with NWJ or LSQ performs poorly, with at best linear gains for exponential increases in sample size. In contrast, if we perform TRE using NWJ or LSQ losses, then we obtain significantly better performance with orders of magnitude fewer samples. These findings are essentially the same as those presented in the main paper for the logistic loss.

5 High-dimensional ratio with large MI toy experiment

In this experiment we estimate the ratio \( p_0/p_m \), where both densities are Gaussian, \( p_0 = \mathcal{N}(0, \Sigma) \) and \( p_m = \mathcal{N}(0, \mathbb{I}) \), where \( \Sigma \) is a block-diagonal covariance matrix, where each block is 2 \( \times \) 2 with 1 on the diagonal and 0.8 on the off-diagonal. Since we know its analytic form, we can view \( p_m \) as a noise distribution, and the ratio-estimation task as an energy-based modelling problem. Alternatively, we may view the problem as a mutual information estimation task, by taking the random variable \( x = (x_1, \ldots, x_d) \sim p_0 \), and defining \( u = (x_1, x_3, \ldots, x_{d-1}) \) and \( v = (x_2, x_4 \ldots x_d) \). By construction, we therefore have \( p(u)p(v) = \mathcal{N}(x; 0, \mathbb{I}) = p_m(x) \).

We generate 100,000 samples for each of the train/validation/test splits. We use a total batch size of 1024, which includes all samples from the waymark trajectories. The bridges in TRE have the form \( \log r_k(x) = x^T W_k x + b_k \), where we enforce that the diagonal entries of \( W_k \) are positive and that the matrix is symmetric. We use the Adam optimiser [8] with an initial learning rate of 0.0001 for TRE, and 0.0005 for single ratio estimation. We use the default Tensorflow settings for \( \beta_1, \beta_2 \) and \( \epsilon \). We train the models for 40,000 iterations, which takes at most 1 hour.

6 MI estimation & representation learning on SpatialMultiOmniglot

We here describe how we created the SpatialMultiOmniglot dataset and give the derivation for the ground truth mutual information values presented in the main paper. We will share the dataset, along with code for the paper, upon publication. We also state the hyperparameter settings used in our experiments.

6.1 Dataset construction

We take the Tensorflow version of the Omniglot dataset [13] and resize it to 28 \( \times \) 28 using the tf.image.resize function. We arrange the data into alphabets \( \{A_i\}_{i=1}^l \), where each alphabet contains \( n_i \) characters. The alphabets are sorted by size, so that \( n_1 > n_2 > \ldots > n_l \). Each character in a alphabet has 20 different

\[1\] The original work from which we borrow this experiment [17] did not provide a detailed explanation or code.
Figure 2: Replica of Figure 1 from the main text, except that we use the NWJ/MINE-f loss \[16, 1\] for both the single ratio estimator & for each ratio in TRE.

\[
\frac{p}{q} = \frac{P}{p_1} \times \frac{P_1}{p_2} \times \frac{P_2}{p_3} \times \frac{P_3}{q}
\]

Figure 3: Replica of Figure 1 from the main text, except that we use the least-square loss from the GAN literature \[12\] for both the single ratio estimator & for each ratio in TRE.

\[
\frac{p}{q} = \frac{P}{p_1} \times \frac{P_1}{p_2} \times \frac{P_2}{p_3} \times \frac{P_3}{q}
\]
Figure 4: Sample efficiency curves for the 1d peaked ratio experiment, using different loss functions.

versions (e.g. there are 20 different images depicting the letter ‘w’). Hence, we can express each alphabet as a set $A_i = \{a_{j,k}^i\}_{k=1}^{n_i}$, where $a_{j,k}^i$ refers to the $k$th version of the $j$th character of the $i$th alphabet.

In order to construct the $d$-dimensional version of the SpatialMultiOmniGlot dataset, we restrict ourselves to the $d$ largest alphabets $\{A_i\}_{i=1}^d$. We then sample a vector of categorical random variables $j = (j_1, \ldots, j_d) \sim \text{Cat}(n_1) \times \ldots \text{Cat}(n_d)$ (15)

where the $i$th categorical distribution is uniform over the set $\{1, \ldots, n_i\}$ and is independent from the other categorical distributions. The vector $j$ should be thought of as an index vector that specifies a particular character from each of the $d$ alphabets.

We then sample two i.i.d random variables $k$ and $k'$, via

$$k = (k_1, \ldots, k_d) \sim \prod_{i=1}^d \text{Cat}(20) \quad k' = (k'_1, \ldots, k'_d) \sim \prod_{i=1}^d \text{Cat}(20)$$ (16)

where, again, each Categorical distribution is uniform over the set $\{1, \ldots, n_i\}$ and is independent from the rest. These vectors should be thought of as index vectors that specify a particular version of a character.

Now, we define a datapoint as a tuple $x = (u, v)$, where

$$u = (a_{j_1,k_1}, \ldots, a_{j_d,k_d}) \quad v = (a_{j_1+1,k'_1}, \ldots, a_{j_d+1,k'_d}).$$ (17)

In words, we construct $u$ and $v$ such that $u_i$ and $v_i$ are consecutive characters within their alphabet (whilst the precise versions of the characters are randomised). Finally, we arrange $u$ and $v$ into a grid using raster ordering. This is possible since we assume $d$ to be a square number.

Importantly, we emphasise that $u, v \in \prod_{i=1}^d A_i$ are discrete random variables defined over a set of template images. They are not defined over a space of pixel values, as is usually the case in image-modelling.

### 6.2 Derivation of ground truth MI

By construction, we have that $u$ and $v$ are conditionally independent given $j$. This means

$$p(u|v,j) = p(u|j).$$ (18)

Furthermore, will assume that, for all $u$ there exists a unique $j_u$ such that

$$p(j_u|u) = 1.$$ (19)
Similarly, for any \( \mathbf{v} \), there exists a unique \( \mathbf{j}_\mathbf{v} \) satisfying the same condition. In words, this simply means that, given a grid of Omniglot images, we assume there is no ambiguity about which characters are present. Using Bayes’ rule, and the fact that for a given \( \mathbf{j}, \mathbf{u} \) is uniquely determined by \( \mathbf{k} \), one can then deduce that

\[
p(\mathbf{u} | \mathbf{j}) = \begin{cases} 
0, & \text{for } j \neq j_u \\
20^{-d}, & \text{for } j = j_u
\end{cases}
\]

and similarly for \( \mathbf{v} \).

We now proceed to derive an analytical formula for the ground truth mutual information between \( \mathbf{u} \) and \( \mathbf{v} \). We show that the mutual information is equal to the sum of the log alphabet sizes

\[
\mathcal{I}(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{d} \log n_i.
\]

This holds because

\[
\mathcal{I}(\mathbf{u}, \mathbf{v}) = \mathbb{E}_{p(\mathbf{u}, \mathbf{v})} \log \frac{p(\mathbf{u}, \mathbf{v})}{p(\mathbf{u})p(\mathbf{v})} = \mathbb{E}_{p(\mathbf{u}, \mathbf{v})} \log \frac{\sum_j p(\mathbf{u}, \mathbf{j} | \mathbf{v})}{\sum_j p(\mathbf{u}, \mathbf{j})} \quad \text{(20)}
\]

We use the Adam optimiser \([8]\) with an initial learning rate of \(10^{-4}\) with default Tensorflow settings for \(\beta_1, \beta_2\) and \(\epsilon\). We gradually decrease the learning rate over the course of training with cosine

\[
\text{6.3 Experimental settings}
\]

We generate 3 versions of the SpatialMultiOmniglot dataset for \(d = 1, 4, 9\). For each version, we sample 50,000 training points and 10,000 validation and test points. As stated in the main text, we use a separable architecture given by

\[
\log r_k(\mathbf{u}, \mathbf{v}) = g(\mathbf{u})^T \mathbf{W}_k f_k(\mathbf{v}),
\]

where \(f_k\) is a convolutional ResNet whose architecture is given in Figure 1a. The function \(g\) is also a convolutional ResNet with almost the same architecture, except that none of its parameters are bridge-specific, and hence the ‘ConditionalScaleShift’ layers simply become ‘ScaleShift’ layers, with no dependence on \(k\).

To construct a mini-batch, we first sample a batch from the joint distribution \(p(\mathbf{u}, \mathbf{v})\). We then obtain samples from \(p(\mathbf{u})p(\mathbf{v})\) by sampling a second batch from the joint distribution (which could overlap with the first batch), and shuffling the \(\mathbf{v}\) vectors across this second batch. Finally, we construct waymark trajectories as described in the main text. For all experiments, the ‘total’ batch size is \(\sim 512\), which includes all samples from the waymark trajectories. Thus, as the number of waymarks increases, the number of trajectories in a batch decreases.

We use the Adam optimiser \([8]\) with an initial learning rate of \(10^{-4}\) with default Tensorflow settings for \(\beta_1, \beta_2\) and \(\epsilon\). We gradually decrease the learning rate over the course of training with cosine
annealing \cite{25}. All models are trained using a single NVIDIA Tesla P100 GPU card for 200,000 iterations, which takes at most a day.

We grid-searched over the type of pooling (max vs. average) and the size of the final dense layer (150n, 300n and 450n, where d = n^2). Interestingly, average pooling was less prone to overfitting and often yielded better final performance, however it was often ‘slow to get started’, with the TRE losses hardly making any progress during the first quarter of training.

For the representation learning evaluations, we first obtained the hidden representations g(u) for the entire dataset. We then trained a collection of independent supervised linear classifiers on top of these representations, in order to predict the alphabetic position of each character in u. We used the L-BFGS optimiser to fit these classifiers via the tfp.optimizer.lbfgs_minimize function, setting the maximum iteration number to 10,000.

7 Energy-based modelling on MNIST

We here discuss the parameterisation of the noise distributions used in the experiments, the exact method for sampling from the learned EBMs, and the experimental settings used for TRE.

For all noise distributions and TRE models, we use the Adam optimiser \cite{6} with an initial learning rate of 10^{-4} with default Tensorflow settings for β_1, β_2 and ε. We gradually decrease the learning rate over the course of training with cosine annealing \cite{25}. All models are trained using a single NVIDIA Tesla P100 GPU card.

7.1 Noise distributions

As stated in the main text, we consider three noise distributions: a multivariate Gaussian, a Gaussian copula and a rational-quadratic neural spline flow (RQ-NSF), all of which are pre-trained via maximum likelihood estimation.

The full-covariance multivariate Gaussian is by far the simplest, and can be fitted in around a minute via np.cov. The Gaussian copula is slightly more complicated. Its density can be written as \( p_{\mathbf{x}} = \mathcal{N}(\{s_1(x_1), \ldots, s_d(x_d); \mu, \Sigma\}) \prod_{i=1}^{d} |s_i'(x_i)|. \) The \( s_i \) are given by the composition of the inverse CDF of a standard normal and the CDF of the univariate \( x_i \). It is possible to exploit this to learn the \( s_i \)—as well as \( \mu \) and \( \Sigma \)—however, we found it slightly simpler to directly parametrise the \( s_i \) via flexible rational-quadratic spline functions \cite{23} of which there are official implementations in Tensorflow and Pytorch and to jointly learn all parameters via maximum likelihood. We follow the basic hyperparameter recommendations in \cite{23}. The hyperparameters that required tuning were the number of bins (we use 128) and the interval widths (which we set to 3 times the standard deviation of the data). For optimisation, we used a batch size of 512 and trained for 40,000 iterations.

Finally, we turn to the RQ-NSF model \cite{23}. We largely adopt the architectural choices of \cite{23}, and so for a more detailed explanation, we refer the reader to their work. We use a multi-scale convolutional architecture comprised of 2 levels, where each level contains 8 ‘steps’. A step consists of an actnorm layer, an invertible 1 \times 1 convolution, and a rational-quadratic coupling transform. The coupling transforms are parameterised by a block of convolution operations following \cite{35}, which use 64 feature maps. The spline functions use 8 bins and the interval width is set to [-3, 3]. We do not ‘factor out’ half of the variables at the end of each level, but do perform ‘squeeze’ operation and an additional 1 \times 1 convolution. For optimisation, we set the batch size to 256, the dropout rate to 0.1, and train for 200,000 iterations, which takes under a day.

7.2 Annealed MCMC Sampling

We here describe how we leverage the specific products-of-experts structure of the TRE model to perform annealed MCMC sampling. Firstly, we initialise a set of MCMC chains with i.i.d samples from the noise distribution \( p_m \). We could then run an MCMC sampler with the full TRE model as the target distribution. However, we instead use an annealing procedure, whereby we iteratively sample from a sequence of distributions that interpolate between \( p_m \) and \( p_0 \). Such distributions can
be obtained by multiplying $p_m$ with an increasing number of bridges

\[ p_k(x) = p_m(x) \prod_{i=k}^{m-1} r_k(x), \quad k = m - 1, \ldots, 0. \]  

(31)

To obtain an even smoother interpolation, we further define exponentially-averaged intermediate distributions

\[ p_{k+1}(x) = p_k(x)^{\beta_k} p_k(x)^{1-\beta_k}, \quad \{\beta_k\} \text{ is a decreasing sequence of numbers ranging from 1 to 0.} \]

In addition to obtaining samples, we can simultaneously use this annealing procedure for estimating the log-likelihood of the model via annealed importance sampling (AIS) \[15\]. We may also run the annealing procedure ‘in reverse’, initialising a chain at a datapoint and iteratively removing bridges until the target distribution of the MCMC sampler is the noise distribution. Using this reverse sampling procedure, we can obtain a second, more conservative, estimate of the log-likelihood via the reverse annealed importance sampling estimator (RAISE) \[2\].

Whilst in principle any MCMC sampler could be used, the efficiency of different samplers can vary greatly. We choose to use the gradient-based No-U-turn sampler (NUTS) \[7\], which is a highly efficient method for many applications. We use the official Tensorflow implementation along with most of the default hyperparameter settings. We set the target acceptance rate to 0.6, and use a max tree depth of 6 during the annealed sampling. We also continue to run the sampler after the annealing phase is finished, using a max tree depth of 10. We use a total of 1000 intermediate distributions with 100 parallel chains.

Finally, recall from the main text that each noise distribution in our experiments can be expressed as invertible transformation $F$ of a standard normal distribution. We use this $F$ to further enhance the efficiency of the NUTS sampler, by performing the sampling in the $z$-space, and then mapping the final results back to $x$-space. Working in $z$-space, by the rules of transformations of random variables, the intermediate distributions of (31) become

\[ p_k(z) = \mathcal{N}(z; 0, I) \prod_{i=k}^{m-1} r_k(F(z)). \]  

(32)

AIS and RAISE can still be applied, just as before, to obtain an estimate of the log-likelihood in $z$-space. The change of variables formula for probability density functions can then be applied to obtain estimated log-likelihoods for the original TRE model in $x$-space. We note that when the noise distribution is a normalising flow, prior work has demonstrated that $z$-space MCMC sampling can be significantly more effective than working in the original data space \[6\].

### 7.3 Experimental settings

We use the standard version of the MNIST dataset \[10\], with 50,000 training points, and 10,000 validation and test points. We follow the same preprocessing steps as \[18\], ‘dequantizing’ the dataset with uniform noise, re-scaling to the unit interval, and then mapping to the real line via a logit transformation.

The architecture for the TRE bridges is given in Figure 1b. The waymark mechanism and associated grid-search is given in Table 1. A consistent observation across all our MNIST experiments was that the first ratio-estimator between the data distribution $p_0$ and a slightly perturbed data distribution $p_1$ was extremely prone to overfitting. We found that the only way to mitigate this problem was to simply drop the ratio by setting the $\alpha_0$ in (2) to a very small value (0.01) rather than exactly 0. Equivalently, this can be viewed as applying standard TRE to a very slightly perturbed data distribution. We note that this perturbation is small enough that is barely visible to the human eye when comparing samples. We conjecture that this problem may stem from the fact that the original MNIST dataset is actually discrete not continuous and the ‘dequantizing’ perturbation used to make the data continuous is perhaps not sufficient.

To form mini-batches, we sample 25 datapoints each from $p_0$ and $p_m$, and then generate waymark trajectories as described in the main text. Thus, the total batch size is $25 \times (m + 1)$. We use the optimisation settings described at the beginning of this section, training for 200,000 iterations, which takes about a day.
7.4 Additional results

In Figure 5 we present a sensitivity analysis showing how the quality of the learned EBM varies as we alter the number of waymarks, as well as the space in which the waymarks are generated. We found that working in $x$-space yielded lower performance compared to working in $z$-space, as measured by the most conservative estimator, RAISE. In particular, we found that the $x$-space mechanism required more waymarks (around 15) to avoid any of the logistic losses saturating close to 0, and it was significantly harder to tune the spacing of the waymarks as indicated by Table 1.

Finally, for the models whose results were given in the main paper, we display extended image samples in Figure 6. Note that these samples are ordered by log-density (lowest density in top left corner, highest in bottom right).

Figure 5: Waymark sensitivity analysis for TRE with copula noise distribution. Both plots show the estimated bits per dimension (bpd) as a function of waymark number. On the left we apply the linear combination waymark mechanism in $z$-space, whilst on the right we apply it in $x$-space. As described in Section 2, we terminate runs where any of the TRE losses saturate close to 0, which is exactly what happened when using the $x$-space mechanism for 5 and 10 waymarks.
Figure 6: Extended MNIST samples
References


