Automatic change-point detection in time series via deep learning

Jie Li[1] , **Paul Fearnhead[2]** , **Piotr Fryzlewicz[1] and Tengyao Wang[1]**

[1]Department of Statistics, London School of Economics and Political Science, London, UK
[2]Department of Mathematics and Statistics, Lancaster University, Lancaster, UK

Address for correspondence: Jie Li, Department of Statistics, London School of Economics and Political Science, Columbia House, Houghton Street, London WC2A 2AE, UK. Email: j.li196@lse.ac.uk

Read before The Royal Statistical Society at the Discussion Meeting on 'Probabilistic and statistical aspects of machine learning' held at the Society's 2023 annual conference in Harrogate on Wednesday, 6 September 2023, the President, Dr Andrew Garrett, in the Chair.

Abstract

Detecting change points in data is challenging because of the range of possible types of change and types of behaviour of data when there is no change. Statistically efficient methods for detecting a change will depend on both of these features, and it can be difficult for a practitioner to develop an appropriate detection method for their application of interest. We show how to automatically generate new offline detection methods based on training a neural network. Our approach is motivated by many existing tests for the presence of a change point being representable by a simple neural network, and thus a neural network trained with sufficient data should have performance at least as good as these methods. We present theory that quantifies the error rate for such an approach, and how it depends on the amount of training data. Empirical results show that, even with limited training data, its performance is competitive with the standard cumulative sum (CUSUM) based classifier for detecting a change in mean when the noise is independent and Gaussian, and can substantially outperform it in the presence of auto-correlated or heavy-tailed noise. Our method also shows strong results in detecting and localizing changes in activity based on accelerometer data.

Keywords: automatic statistician; classification; likelihood-free inference; neural networks; structural breaks; supervised learning

1 Introduction

Detecting change points in data sequences is of interest in many application areas such as bioinformatics (Picard et al., 2005), climatology (Reeves et al., 2007), signal processing (Haynes et al., 2017), and neuroscience (Oh et al., 2005). In this work, we are primarily concerned with the problem of offline change-point detection, where the entire data is available to the analyst beforehand. Over the past few decades, various methodologies have been extensively studied in this area, see Killick et al. (2012), Jandhyala et al. (2013), Fryzlewicz (2014, 2023), Wang and Samworth (2018), Truong et al. (2020) and references therein. Most research on change-point detection has concentrated on detecting and localizing different types of change, e.g. change in mean (Fryzlewicz, 2014; Killick et al., 2012), variance (Gao et al., 2019; Li et al., 2015), median (Fryzlewicz, 2021), or slope (Baranowski et al., 2019; Fearnhead et al., 2019), amongst many others.

Many change-point detection methods are based upon modelling data when there is no change and when there is a single change, and then constructing an appropriate test statistic to detect the presence of a change (e.g. Fearnhead & Rigaill, 2020; James et al., 1987). The form of a good test statistic will vary with our modelling assumptions and the type of change we wish to detect. This can lead to difficulties in practice. As we use new models, it is unlikely

that there will be a change-point detection method specifically designed for our modelling assumptions. Furthermore, developing an appropriate method under a complex model may be challenging, while in some applications an appropriate model for the data may be unclear but we may have substantial historical data that shows what patterns of data to expect when there is, or is not, a change.

In these scenarios, currently a practitioner would need to choose the existing change detection method which seems the most appropriate for the type of data they have and the type of change they wish to detect. To obtain reliable performance, they would then need to adapt its implementation, for example tuning the choice of threshold for detecting a change. Often, this would involve applying the method to simulated or historical data.

To address the challenge of automatically developing new change detection methods, this paper is motivated by the question: Can we construct new test statistics for detecting a change based only on having labelled examples of change points? We show that this is indeed possible by training a neural network to classify whether or not a dataset has a change of interest. This turns change-point detection in a supervised learning problem.

A key motivation for our approach are results that show many common test statistics for detecting changes, such as the CUSUM test for detecting a change in mean, can be represented by simple neural networks. This means that with sufficient training data, the classifier learnt by such a neural network will give performance at least as good as classifiers corresponding to these standard tests. In scenarios where a standard test, such as CUSUM, is being applied but its modelling assumptions do not hold, we can expect the classifier learnt by the neural network to outperform it.

There has been increasing recent interest in whether ideas from machine learning, and methods for classification, can be used for change-point detection. Within computer science and engineering, these include a number of methods designed for and that show promise on specific applications (e.g. Ahmadzadeh, 2018; De Ryck et al., 2021; Gupta et al., 2022; Huang et al., 2023). Within statistics, Londschien et al. (2022) and Lee et al. (2023) consider training a classifier as a way to estimate the likelihood-ratio statistic for a change. However, these methods train the classifier in an unsupervised way on the data being analysed, using the idea that a classifier would more easily distinguish between two segments of data if they are separated by a change point. Chang et al. (2019) use simulated data to help tune a kernel-based change-detection method. Methods that use historical, labelled data have been used to train the tuning parameters of change-point algorithms (e.g. Hocking et al., 2015; Liehrmann et al., 2021). Also, neural networks have been employed to construct similarity scores of new observations to learned pre-change distributions for online change-point detection (Lee et al., 2023). However, we are unaware of any previous work using historical, labelled data to develop offline change-point methods. As such, and for simplicity, we focus on the most fundamental aspect, namely the problem of detecting a single change. Detecting and localizing multiple changes is considered in Section 6 when analysing activity data. We remark that by viewing the change-point detection problem as a classification instead of a testing problem, we aim to control the overall mis-classification error rate (MER) instead of handling the Type I and Type II errors separately. In practice, asymmetric treatment of the two error types can be achieved by suitably re-weighting mis-classification in the two directions in the training loss function.

The method we develop has parallels with likelihood-free inference methods (Beaumont, 2019; Gourieroux et al., 1993) in that one application of our work is to use the ability to simulate from a model so as to circumvent the need to analytically calculate likelihoods. However, the approach we take is very different from standard likelihood-free methods which tend to use simulation to estimate the likelihood function itself. By comparison, we directly target learning a function of the data that can discriminate between instances that do or do not contain a change (though see Gutmann et al., 2018 for likelihood-free methods based on re-casting the likelihood as a classification problem).

For an introduction to the statistical aspects of neural network-based classification, albeit not specifically in a change-point context, see Ripley (1994).

We now briefly introduce our notation. For any $n \in \mathbb{Z}^+$, we define $[n] := \{1, \ldots, n\}$. We take all vectors to be column vectors unless otherwise stated. Let $\mathbf{1}_n$ be the all-one vector of length n. Let $\mathbb{1}\{\cdot\}$ represent the indicator function. The vertical symbol $|\cdot|$ represents the absolute value or cardinality of $\cdot$ depending on the context. For vector $\boldsymbol{x} = (x_1, \ldots, x_n)^\top$, we define its p-norm as $\|\boldsymbol{x}\|_p := \left(\sum_{i=1}^n |x_i|^p\right)^{1/p}$, $p \geq 1$; when $p = \infty$, define $\|\boldsymbol{x}\|_\infty := \max_i |x_i|$. All proofs, as well as additional simulations and real data analyses appear in the online supplementary material.
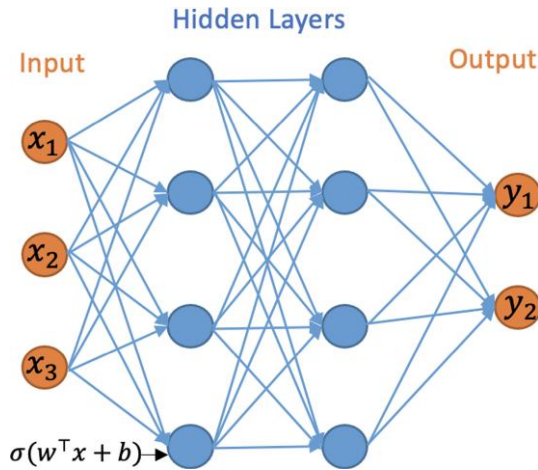
Figure 1. A neural network with two hidden layers and width vector $\boldsymbol{m} = (4, 4)$.

2 Neural networks

The initial focus of our work is on the binary classification problem for whether a change point exists in a given time series. We will work with multi-layer neural networks with Rectified Linear Unit (ReLU) activation functions and binary output. The multi-layer neural network consists of an input layer, hidden layers, and an output layer, and can be represented by a directed acyclic graph, see Figure 1. Let $L \in \mathbb{Z}^+$ represent the number of hidden layers and $\boldsymbol{m} = (m_1, \ldots, m_L)^\top$ the vector of the hidden layer widths, i.e. m_i is the number of nodes in the ith hidden layer. For a neural network with L hidden layers, we use the convention that $m_0 = n$ and $m_{L+1} = 1$. For any bias vector $\boldsymbol{b} = (b_1, b_2, \ldots, b_r)^\top \in \mathbb{R}^r$, define the shifted activation function $\sigma_b : \mathbb{R}^r \to \mathbb{R}^r$:

$$\sigma_b((y_1, \ldots, y_r)^\top) = (\sigma(y_1 - b_1), \ldots, \sigma(y_r - b_r))^\top,$$

where $\sigma(x) = \max(x, 0)$ is the ReLU activation function. The neural network can be mathematically represented by the composite function $h : \mathbb{R}^n \to \{0, 1\}$ as

$$h(\boldsymbol{x}) := \sigma_\lambda^* W_L \sigma_{b_L} W_{L-1} \sigma_{b_{L-1}} \cdots W_1 \sigma_{b_1} W_0 \boldsymbol{x}, \tag{1}$$

where $\sigma_\lambda^*(x) = \mathbb{1}\{x > \lambda\}$, $\lambda > 0$ and $W_\ell \in \mathbb{R}^{m_{\ell+1} \times m_\ell}$ for $\ell \in \{0, \ldots, L\}$ represent the weight matrices. We define the function class $\mathcal{H}_{L,\boldsymbol{m}}$ to be the class of functions $h(\boldsymbol{x})$ with L hidden layers and width vector $\boldsymbol{m}$.

The output layer in equation (1) employs the shifted heaviside function $\sigma_\lambda^*(x)$, which is used for binary classification as the final activation function. This choice is guided by the fact that we use the 0–1 loss, which focuses on the percentage of samples assigned to the correct class, a natural performance criterion for binary classification. Besides its wide adoption in machine learning practice, another advantage of using the 0–1 loss is that it is possible to utilise the theory of the Vapnik–Chervonenkis (VC) dimension (see, e.g. Shalev-Shwartz & Ben-David, 2014, Definition 6.5) to bound the generalization error of a binary classifier equipped with this loss; indeed, this is the approach we take in this work. The relevant results regarding the VC dimension of neural network classifiers are, e.g. in Bartlett et al. (2019). As in Schmidt-Hieber (2020), we work with the exact minimizer of the empirical risk. In both binary or multi-class classification, it is possible to work with other losses which make it computationally easier to minimise the corresponding risk, see, e.g. Bos and Schmidt-Hieber (2022), who use a version of the cross-entropy loss. However, loss functions different from the 0–1 loss make it impossible to use VC-dimension arguments to control the generalization error, and more involved arguments, such as those using the covering number (Bos & Schmidt-Hieber, 2022) need to be used instead. We do not pursue these generalizations in the current work.

3 CUSUM-based classifier and its generalizations are neural networks

3.1 Change in mean

We initially consider the case of a single change point with an unknown location $\tau \in [n-1], n \geq 2$, in the model

$$X = \mu + \xi,$$
$$\mu = (\mu_L \mathbb{1}\{i \leq \tau\} + \mu_R \mathbb{1}\{i > \tau\})_{i \in [n]} \in \mathbb{R}^n,$$

where μ_L, μ_R are the unknown signal values before and after the change point; $\xi \sim N_n(0, I_n)$. The CUSUM test is widely used to detect mean changes in univariate data. For the observation x, the CUSUM transformation $\mathcal{C}: \mathbb{R}^n \to \mathbb{R}^{n-1}$ is defined as $\mathcal{C}(x) := (v_1^\top x, \ldots, v_{n-1}^\top x)^\top$, where $v_i := (\sqrt{\frac{n-i}{in}}\mathbf{1}_i^\top, -\sqrt{\frac{i}{(n-i)n}}\mathbf{1}_{n-i}^\top)^\top$ for $i \in [n-1]$. Here, for each $i \in [n-1]$, $(v_i^\top x)^2$ is the log likelihood-ratio statistic for testing a change at time i against the null of no change (e.g. Baranowski et al., 2019). For a given threshold $\lambda > 0$, the classical CUSUM test for a change in the mean of the data is defined as

$$h_\lambda^{\text{CUSUM}}(x) = \mathbb{1}\{\|\mathcal{C}(x)\|_\infty > \lambda\}.$$

The following lemma shows that $h_\lambda^{\text{CUSUM}}(x)$ can be represented as a neural network.

> **Lemma 3.1** For any $\lambda > 0$, we have $h_\lambda^{\text{CUSUM}}(x) \in \mathcal{H}_{1,2n-2}$.

The fact that the widely used CUSUM statistic can be viewed as a simple neural network has far-reaching consequences: this means that given enough training data, a neural network architecture that permits the CUSUM-based classifier as its special case cannot do worse than CUSUM in classifying change-point vs. no-change-point signals. This serves as the main motivation for our work, and a prelude to our next results.

3.2 Beyond the mean change model

We can generalise the simple change in mean model to allow for different types of change or for non-independent noise. In this section, we consider change-point models that can be expressed as a change in regression problem, where the model for data given a change at τ is of the form

$$X = Z\beta + c_\tau \phi + \Gamma\xi, \tag{2}$$

where for some $p \geq 1$, Z is an $n \times p$ matrix of covariates for the model with no change, c_τ is an $n \times 1$ vector of covariates specific to the change at τ, and the parameters β and ϕ are, respectively, a $p \times 1$ vector and a scalar. The noise is defined in terms of an $n \times n$ matrix Γ and an $n \times 1$ vector of independent standard normal random variables, ξ.

For example, the change in mean problem has $p = 1$, with Z a column vector of ones, and c_τ being a vector whose first τ entries are zeros, and the remaining entries are ones. In this formulation, β is the pre-change mean and ϕ is the size of the change. The change in slope problem (Fearnhead et al., 2019) has $p = 2$ with the columns of Z being a vector of ones, and a vector whose ith entry is i; and c_τ has ith entry that is $\max\{0, i - \tau\}$. In this formulation, β defines the pre-change linear mean and ϕ the size of the change in slope. Choosing Γ to be proportional to the identity matrix gives a model with independent, identically distributed noise; but other choices would allow for auto-correlation.

The following result is a generalization of Lemma 3.1, which shows that the likelihood-ratio test for equation (2), viewed as a classifier, can be represented by our neural network.

> **Lemma 3.2** Consider the change-point model (2) with a possible change at $\tau \in [n-1]$. Assume further that Γ is invertible. Then there is an $h^* \in \mathcal{H}_{1,2n-2}$ equivalent to the likelihood-ratio test for testing $\phi = 0$ against $\phi \neq 0$.

Importantly, this result shows that for this much wider class of change-point models, we can replicate the likelihood-ratio-based classifier for change using a simple neural network.

Other types of changes can be handled by suitably pre-transforming the data. For instance, squaring the input data would be helpful in detecting changes in the variance and if the data followed an AR(1) structure, then changes in auto-correlation could be handled by including transformations of the original input of the form $(x_t x_{t+1})_{t=1,\ldots,n-1}$. On the other hand, even if such transformations are not supplied as the input, a neural network of suitable depth is able to approximate these transformations and consequently successfully detect the change (Schmidt-Hieber, 2020, Lemma A.2). This is illustrated in Figure S3 of the online supplementary material, where we compare the performance of neural network-based classifiers of various depths constructed with and without using the transformed data as inputs.

4 Generalization error of neural network change-point classifiers

In Section 3, we showed that CUSUM and generalised CUSUM could be represented by a neural network. Therefore, with a large enough amount of training data, a trained neural network classifier that included CUSUM, or generalised CUSUM, as a special case, would perform no worse than it on unseen data. In this section, we provide generalization bounds for a neural network classifier for the change-in-mean problem, given a finite amount of training data. En route to this main result, stated in Theorem 4.3, we provide generalization bounds for the CUSUM-based classifier, in which the threshold has been chosen on a finite training dataset.

We write $P(n, \tau, \mu_L, \mu_R)$ for the distribution of the multivariate normal random vector $X \sim N_n(\boldsymbol{\mu}, I_n)$, where $\boldsymbol{\mu} := (\mu_L \mathbb{1}\{i \leq \tau\} + \mu_R \mathbb{1}\{i > \tau\})_{i \in [n]}$. Define $\eta := \tau/n$. Lemma 4.1 and Corollary 4.1 control the mis-classification error of the CUSUM-based classifier.

Lemma 4.1   Fix $\varepsilon \in (0, 1)$. Suppose $X \sim P(n, \tau, \mu_L, \mu_R)$ for some $\tau \in \mathbb{Z}^+$ and $\mu_L, \mu_R \in \mathbb{R}$.

(a) If $\mu_L = \mu_R$, then $\mathbb{P}\{\|\mathcal{C}(X)\|_\infty > \sqrt{2 \log(n/\varepsilon)}\} \leq \varepsilon$.
(b) If $|\mu_L - \mu_R|\sqrt{\eta(1-\eta)} > \sqrt{8 \log(n/\varepsilon)/n}$, then $\mathbb{P}\{\|\mathcal{C}(X)\|_\infty \leq \sqrt{2 \log(n/\varepsilon)}\} \leq \varepsilon$.

For any $B > 0$, define

$$\Theta(B) := \left\{ (\tau, \mu_L, \mu_R) \in [n-1] \times \mathbb{R} \times \mathbb{R} : |\mu_L - \mu_R|\sqrt{\tau(n-\tau)}/n \in \{0\} \cup (B, \infty) \right\}.$$

Here, $|\mu_L - \mu_R|\sqrt{\tau(n-\tau)}/n = |\mu_L - \mu_R|\sqrt{\eta(1-\eta)}$ can be interpreted as the signal-to-noise ratio (SNR) of the mean change problem. Thus, $\Theta(B)$ is the parameter space of data distributions where there is either no change or a single change point in mean whose SNR is at least B. The following corollary controls the mis-classification risk of a CUSUM statistics-based classifier:

Corollary 4.1   Fix $B > 0$. Let π_0 be any prior distribution on $\Theta(B)$, then draw $(\tau, \mu_L, \mu_R) \sim \pi_0$ and $X \sim P(n, \tau, \mu_L, \mu_R)$, and define $Y = \mathbb{1}\{\mu_L \neq \mu_R\}$. For $\lambda = B\sqrt{n}/2$, the classifier $h_\lambda^{\mathrm{CUSUM}}$ satisfies

$$\mathbb{P}(h_\lambda^{\mathrm{CUSUM}}(X) \neq Y) \leq n e^{-nB^2/8}.$$

Theorem 4.2 below, which is based on Corollary 4.1, Bartlett et al. (2019, Theorem 7) and Mohri et al. (2012, Corollary 3.4), shows that the empirical risk minimizer in the neural network class $\mathcal{H}_{1,2n-2}$ has good generalization properties over the class of change-point problems parameterised by $\Theta(B)$. Given training data $(X^{(1)}, Y^{(1)}), \ldots, (X^{(N)}, Y^{(N)})$ and any $h : \mathbb{R}^n \to \{0, 1\}$, we define the empirical risk of h as

$$L_N(h) := \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}\{Y^{(i)} \neq h(X^{(i)})\}.$$
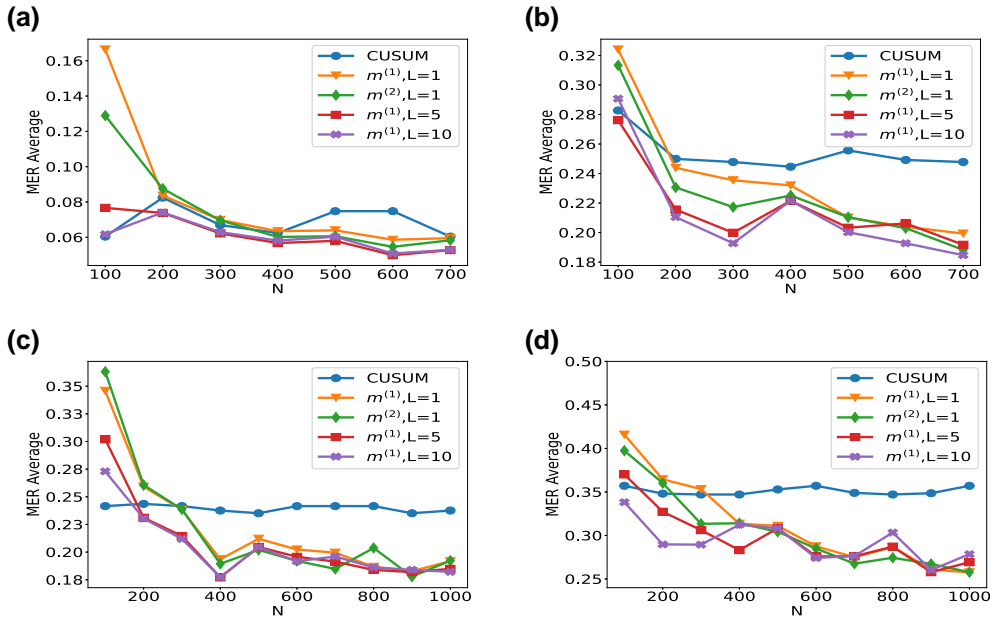
Figure 2. Plot of the test set mis-classification error rate, computed on a test set of size $N_{\text{test}} = 30{,}000$, against training sample size N for detecting the existence of a change point on data series of length $n = 100$. We compare the performance of the CUSUM test and neural networks from four function classes: $\mathcal{H}_{1,m^{(1)}}$, $\mathcal{H}_{1,m^{(2)}}$, $\mathcal{H}_{5,m^{(1)}\mathbf{1}_5}$, and $\mathcal{H}_{10,m^{(1)}\mathbf{1}_{10}}$, where $m^{(1)} = 4\lfloor \log_2(n) \rfloor$ and $m^{(2)} = 2n - 2$, respectively, under scenarios S1, S1 ′, S2, and S3 described in Section 5. (a) Scenario S1 with $\rho_t = 0$. (b) Scenario S1 ′ with $\rho_t = 0.7$. (c) Scenario S2 with $\rho_t \sim \text{Unif}([0, 1])$. (d) Scenario S3 with Cauchy noise.

Theorem 4.2 Fix $B > 0$ and let π_0 be any prior distribution on $\Theta(B)$. We draw $(\tau, \mu_L, \mu_R) \sim \pi_0$, $X \sim P(n, \tau, \mu_L, \mu_R)$, and set $Y = \mathbb{1}\{\mu_L \neq \mu_R\}$. Suppose that the training data $\mathcal{D} := ((X^{(1)}, Y^{(1)}), \ldots, (X^{(N)}, Y^{(N)}))$ consist of independent copies of (X, Y) and $h_{\text{ERM}} := \arg\ \min_{h \in \mathcal{H}_{1,2n-2}} L_N(h)$ is the empirical risk minimizer. There exists a universal constant $C > 0$ such that for any $\delta \in (0, 1)$, equation (3) holds with probability $1 - \delta$.

$$\mathbb{P}(h_{\text{ERM}}(X) \neq Y \mid \mathcal{D}) \leq ne^{-nB^2/8} + C\sqrt{\frac{n^2 \log(n) \log(N) + \log(1/\delta)}{N}}. \quad (3)$$

The theoretical results derived for the neural network-based classifier, here and below, all rely on the fact that the training and test data are drawn from the same distribution. However, we observe that in practice, even when the training and test sets have different error distributions, neural network-based classifiers still provide accurate results on the test set; see our discussion of Figure 2 in Section 5 for more details. The mis-classification error in equation (3) is bounded by two terms. The first term represents the mis-classification error of CUSUM-based classifier, see Corollary 4.1, and the second term depends on the complexity of the neural network class measured in its VC dimension. Theorem 4.2 suggests that for training sample size $N \gg n^2 \log n$, a well-trained single hidden layer neural network with $2n - 2$ hidden nodes would have comparable performance to that of the CUSUM-based classifier. However, as we will see in Section 5, in practice, a much smaller training sample size N is needed for the neural network to be competitive in the change-point detection task. This is because the $2n - 2$ hidden layer nodes in the neural network representation of h_λ^{CUSUM} encode the components of the CUSUM transformation $(\pm v_t^\top x : t \in [n-1])$, which are highly correlated.

By suitably pruning the hidden layer nodes, we can show that a single hidden layer neural network with $O(\log n)$ hidden nodes is able to represent a modified version of the CUSUM-based classifier with essentially the same mis-classification error. More precisely, let $Q := \lfloor \log_2(n/2) \rfloor$ and write $T_0 := \{2^q : 0 \leq q \leq Q\} \cup \{n - 2^q : 0 \leq q \leq Q\}$. We can then define

$$h_{\lambda^*}^{\mathrm{CUSUM}_*}(\boldsymbol{X}) = \mathbb{1}l\{\max_{t \in T_0}|\boldsymbol{v}_t^{\mathsf{T}}\boldsymbol{X}| > \lambda^*\}.$$

By the same argument as in Lemma 3.1, we can show that $h_{\lambda^*}^{\mathrm{CUSUM}_*} \in \mathcal{H}_{1,4\lfloor \log_2{(n)}\rfloor}$ for any $\lambda^* > 0$. The following theorem shows that high classification accuracy can be achieved under a weaker training sample size condition compared to Theorem 4.2.

Theorem 4.3 Fix $B > 0$ and let the training data $\mathcal{D}$ be generated as in Theorem 4.2. Let $h_{\mathrm{ERM}} := \arg\ \min_{h \in \mathcal{H}_{L,m}} L_N(h)$ be the empirical risk minimizer for a neural network with $L \geq 1$ layers and $\boldsymbol{m} = (m_1, \ldots, m_L)^{\mathsf{T}}$ hidden layer widths. If $m_1 \geq 4\lfloor \log_2{(n)}\rfloor$ and $m_r m_{r+1} = O(n \log n)$ for all $r \in [L-1]$, then there exists a universal constant $C > 0$ such that for any $\delta \in (0, 1)$, equation (4) holds with probability $1 - \delta$.

$$\mathbb{P}(h_{\mathrm{ERM}}(\boldsymbol{X}) \neq Y \mid \mathcal{D}) \leq 2\lfloor \log_2{(n)}\rfloor e^{-nB^2/24}$$
$$+ C\sqrt{\frac{L^2 n \log^2{(Ln)}\log{(N)} + \log{(1/\delta)}}{N}}. \tag{4}$$

Theorem 4.3 generalises the single hidden layer neural network representation in Theorem 4.2 to multiple hidden layers. In practice, multiple hidden layers help to keep the MER low even when *N* is small, see Section 5. Theorems 4.2 and 4.3 are examples of how to derive generalization errors of a neural network-based classifier in the change-point detection task. The same workflow can be employed in other types of changes, provided that suitable representation results of likelihood-based tests in terms of neural networks (e.g. Lemma 3.2) can be obtained. In a general result of this type, the generalization error of the neural network will again be bounded by a sum of the error of the likelihood-based classifier together with a term originating from the VC-dimension bound of the complexity of the neural network architecture.

We further remark that for simplicity of discussion, we have focused our attention on data models where the noise vector $\boldsymbol{\xi} = \boldsymbol{X} - \mathbb{E}\boldsymbol{X}$ has independent and identically distributed normal components. However, since CUSUM-based tests are available for temporally correlated or sub-Weibull data, with suitably adjusted test threshold values, the above theoretical results readily generalise to such settings. See Theorems S4 and S6 in the online supplementary material for more details.

5 Numerical study

We now investigate empirically our approach of learning a change-point detection method by training a neural network. Motivated by the results from the previous section, we will fit a neural network with a single layer and consider how varying the number of hidden layers and the amount of training data affects performance. We will compare to a test based on the CUSUM statistic, both for scenarios where the noise is independent and Gaussian, and for scenarios where there is auto-correlation or heavy-tailed noise. The CUSUM test can be sensitive to the choice of threshold, particularly when we do not have independent Gaussian noise, so we tune its threshold based on training data.

When training the neural network, we first standardise the data onto $[0, 1]$, i.e. $\tilde{x}_i = ((x_{ij} - x_i^{\min})/(x_i^{\max} - x_i^{\min}))_{j \in [n]}$, where $x_i^{\max} := \max_j x_{ij}$, $x_i^{\min} := \min_j x_{ij}$. This makes the neural network procedure invariant to either adding a constant to the data or scaling the data by a constant, which are natural properties to require. We train the neural network by minimizing the cross-entropy loss on the training data. We run training for 200 epochs with a batch size of 32 and a learning rate of 0.001 using the Adam optimizer (Kingma & Ba, 2015). These hyperparameters are chosen based on a training dataset with cross-validation, more details can be found in Section 2 of the online supplementary material.

We generate our data as follows. Given a sequence of length *n*, we draw $\tau \sim \mathrm{Unif}\{2, \ldots, n-2\}$, set $\mu_L = 0$ and draw $\mu_R|\tau \sim \mathrm{Unif}([-1.5b, -0.5b] \cup [0.5b, 1.5b])$, where $b := \sqrt{\frac{8n \log{(20n)}}{\tau(n-\tau)}}$ is chosen

in line with Lemma 4.1 to ensure a good range of SNRs. We then generate $\boldsymbol{x}_1 = (\mu_\mathrm{L} \mathbb{1}_{\{t \leq \tau\}} + \mu_\mathrm{R} \mathbb{1}_{\{t > \tau\}} + \varepsilon_t)_{t \in [n]}$, with the noise $(\varepsilon_t)_{t \in [n]}$ following an AR(1) model with possibly time-varying auto-correlation $\varepsilon_t | \rho_t = \xi_1$ for $t = 1$ and $\rho_t \varepsilon_{t-1} + \xi_t$ for $t \geq 2$, where $(\xi_t)_{t \in [n]}$ are independent, possibly heavy-tailed noise. The auto-correlations ρ_t and innovations ξ_t are from one of the four scenarios:

S1: $n = 100$, $N \in \{100, 200, \ldots, 700\}$, $\rho_t = 0$, and $\xi_t \sim N(0, 1)$.
S1′: $n = 100$, $N \in \{100, 200, \ldots, 700\}$, $\rho_t = 0.7$, and $\xi_t \sim N(0, 1)$.
S2: $n = 100$, $N \in \{100, 200, \ldots, 1,000\}$, $\rho_t \sim \mathrm{Unif}([0, 1])$, and $\xi_t \sim N(0, 2)$.
S3: $n = 100$, $N \in \{100, 200, \ldots, 1,000\}$, $\rho_t = 0$, and $\xi_t \sim \mathrm{Cauchy}(0, 0.3)$.

The above procedure is then repeated $N/2$ times to generate independent sequences $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N/2}$ with a single change, and the associated labels are $(y_1, \ldots, y_{N/2})^\top = \mathbf{1}_{N/2}$. We then repeat the process another $N/2$ times with $\mu_\mathrm{R} = \mu_\mathrm{L}$ to generate sequences without changes $\boldsymbol{x}_{N/2+1}, \ldots, \boldsymbol{x}_N$ with $(y_{N/2+1}, \ldots, y_N)^\top = \mathbf{0}_{N/2}$. The data with and without change $(\boldsymbol{x}_i, y_i)_{i \in [N]}$ are combined and randomly shuffled to form the training data. The test data are generated in a similar way, with a sample size $N_\text{test} = 30{,}000$ and the slight modification that $\mu_\mathrm{R} | \tau \sim \mathrm{Unif}([-1.75b, -0.25b] \cup [0.25b, 1.75b])$ when a change occurs. We note that the test data is drawn from the same distribution as the training set, though potentially having changes with SNRs outside the range covered by the training set. We have also conducted robustness studies to investigate the effect of training the neural networks on scenario S1 and test on S1′, S2, or S3. Qualitatively similar results to Figure 2 have been obtained in this mis-specified setting (see Figure S2 of the online supplementary material). We compare the performance of the CUSUM-based classifier with the threshold cross-validated on the training data with neural networks from four function classes: $\mathcal{H}_{1, m^{(1)}}$, $\mathcal{H}_{1, m^{(2)}}$, $\mathcal{H}_{5, m^{(1)} \mathbf{1}_5}$ and $\mathcal{H}_{10, m^{(1)} \mathbf{1}_{10}}$, where $m^{(1)} = 4\lfloor \log_2(n) \rfloor$ and $m^{(2)} = 2n - 2$, respectively (cf. Theorem 4.3 and Lemma 3.1). Figure 2 shows the test MER of the four procedures in the four scenarios S1, S1′, S2, and S3. We observe that when data are generated with independent Gaussian noise (Figure 2a), the trained neural networks with $m^{(1)}$ and $m^{(2)}$ single hidden layer nodes attain very similar test MER compared to the CUSUM-based classifier. This is in line with our Theorem 4.3. More interestingly, when noise has either auto-correlation (Figure 2b and c) or heavy-tailed distribution (Figure 2d), trained neural networks with $(L, \mathbf{m})$: $(1, m^{(1)})$, $(1, m^{(2)})$, $(5, m^{(1)} \mathbf{1}_5)$, and $(10, m^{(1)} \mathbf{1}_{10})$ outperform the CUSUM-based classifier, even after we have optimised the threshold choice of the latter. In addition, as shown in Figure S1 in the online supplementary material, when the first two layers of the network are set to carry out truncation, which can be seen as a composition of two ReLU operations, the resulting neural network outperforms the Wilcoxon statistics-based classifier (Dehling et al., 2015), which is a standard benchmark for change-point detection in the presence of heavy-tailed noise. Furthermore, from Figure 2, we see that increasing L can significantly reduce the average MER when $N \leq 200$. Theoretically, as the number of layers L increases, the neural network is better able to approximate the optimal decision boundary, but it becomes increasingly difficult to train the weights due to issues such as vanishing gradients (He et al., 2016). A combination of these considerations leads us to develop deep neural network architecture with residual connections for detecting multiple changes and multiple change types in Section 6.

6 Detecting multiple changes and multiple change types—case study

From the previous section, we see that single and multiple hidden layer neural networks can represent CUSUM or generalised CUSUM tests and may perform better than likelihood-based test statistics when the model is mis-specified. This prompted us to seek a general network architecture that can detect, and even classify, multiple types of change. Motivated by the similarities between signal processing and image recognition, we employed a deep convolutional neural network (CNN) (Yamashita et al., 2018) to learn the various features of multiple change types. However, stacking more CNN layers cannot guarantee a better network because of vanishing gradients in training (He et al., 2016). Therefore, we adopted the residual block structure (He et al., 2016) for our neural network architecture. After experimenting with various architectures with

Table 1. Test classification accuracy of oracle likelihood-ratio-based method (LR^oracle), adaptive likelihood-ratio method (LR^adapt), and our residual neural network (NN) classifier for set-ups with weak and strong signal-to-noise ratios (SNRs)

	Weak SNR			Strong SNR		
	LR^oracle	LR^adapt	NN	LR^oracle	LR^adapt	NN
Class 1	0.9787	0.9457	0.8062	0.9787	0.9341	0.9651
Class 2	0.8443	0.8164	0.8882	1.0000	0.7784	0.9860
Class 3	0.8350	0.8291	0.8585	0.9902	0.9902	0.9705
Class 4	0.9960	0.9453	0.8826	0.9980	0.9372	0.9312
Class 5	0.8729	0.8604	0.8353	0.9958	0.9917	0.9147
Accuracy	0.9056	0.8796	0.8660	0.9924	0.9260	0.9672

Note. Data are generated as a mixture of no change point in mean or variance (Class 1), change in mean only (Class 2), change in variance only (Class 3), no-change in a non-zero slope (Class 4), and change in slope only (Class 5). We report the true positive rate of each class and the accuracy in the last row.

different numbers of residual blocks and fully connected layers on synthetic data, we arrived at a network architecture with 21 residual blocks followed by a number of fully connected layers. Figure S5 of the online supplementary material shows an overview of the architecture of the final general-purpose deep neural network for change-point detection. The precise architecture and training methodology of this network $\widehat{NN}$ can be found in Section 3 of the online supplementary material. Neural architecture search approaches (see Paaß & Giesselbach, 2023, Section 2.4.3) offer principled ways of selecting neural architectures. Some of these approaches could be made applicable in our setting.

We demonstrate the power of our general purpose change-point detection network in a numerical study. We train the network on $N = 10,000$ instances of data sequences generated from a mixture of no change point in mean or variance, change in mean only, change in variance only, no-change in a non-zero slope, and change in slope only, and compare its classification performance on a test set of size 2,500 against that of oracle likelihood-based classifiers (where we pre-specify whether we are testing for change in mean, variance or slope) and adaptive likelihood-based classifiers (where we combine likelihood-based tests using the Bayesian information criterion). Details of the data-generating mechanism and classifiers can be found in Section 2 of the online supplementary material. The classification accuracy of the three approaches in weak and strong SNR settings is reported in Table 1. We see that the neural network-based approach achieves similar classification accuracy as adaptive likelihood-based method for weak SNR and higher classification accuracy than the adaptive likelihood-based method for strong SNR. We would not expect the neural network to outperform the oracle likelihood-based classifiers, as it has no knowledge of the exact change type of each time series.

We now consider an application to detecting different types of change. The HASC (Human Activity Sensing Consortium) project data contain motion sensor measurements during a sequence of human activities, including 'stay', 'walk', 'jog', 'skip', 'stair up', and 'stair down'. Complex changes in sensor signals occur during transition from one activity to the next (see Figure 3). We have 28 labels in HASC data, see Figure S6 of the online supplementary material. To agree with the dimension of the output, we drop two dense layers 'Dense(10)' and 'Dense(20)' in Figure S5 of the online supplementary material. The resulting network can be effectively applied for change-point detection in sensory signals of human activities and can achieve high accuracy in change-point classification tasks (Figure S8 of the online supplementary material).

Finally, we remark that our neural network-based change-point detector can be utilised to detect multiple change points. Algorithm 1 outlines a general scheme for turning a change-point classifier into a location estimator, where we employ an idea similar to that of MOSUM (Eichinger & Kirch, 2018) and repeatedly apply a classifier ψ to data from a sliding window of size n. Here, we require ψ applied to each data segment $X^*_{[i,i+n]}$ to output both the class label $L_i = 0$ or 1 if no change or a change is predicted and the corresponding probability p_i of having a change. In our particular example, for each data segment $X^*_{[i,i+n]}$ of length $n = 700$, we define $\psi(X^*_{[i,i+n]}) = 0$ if $\widehat{NN}(X^*_{[i,i+n]})$
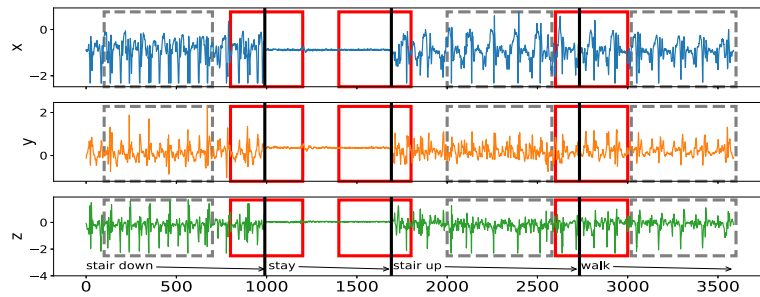
Figure 3. The sequence of accelerometer data in *x*, *y*, and *z* axes. From left to right, there are four activities: 'stair down', 'stay', 'stair up', and 'walk', their change points are 990, 1,691, 2,733, respectively marked by black solid lines. The grey rectangles represent the group of 'no-change' with labels: 'stair down', 'stair up', and 'walk'. The red rectangles represent the group of 'one change' with labels: 'stair down →stay', 'stay →stair up', and 'stair up →walk'.

Algorithm 1 Algorithm for change-point localization

Input: new data $x_1^*, \ldots, x_{n^*}^* \in \mathbb{R}^d$, a trained classifier $\psi \colon \mathbb{R}^{d \times n} \to \{0, 1\}, \gamma > 0$.

1 Form $X_{[i,i+n)}^* := (x_i^*, \ldots, x_{i+n-1})$ and compute $L_i \leftarrow \psi(X_{[i,i+n)}^*)$ for all $i = 1, \ldots, n^* - n + 1$;

2 Compute $\bar{L}_i \leftarrow n^{-1} \sum_{j=i-n+1}^{i} L_j$ for $i = n, \ldots, n^* - n + 1$;

3 Let $\{[s_1, e_1], \ldots, [s_{\hat{v}}, e_{\hat{v}}]\}$ be the set of all maximal segments such that $\bar{L}_i \geq \gamma$ for all $i \in [s_r, e_r], r \in [\hat{v}]$;

4 Compute $\hat{\tau}_r \leftarrow \arg\max_{i \in [s_r, e_r]} \bar{L}_i$ for all $r \in [\hat{v}]$;

Output: Estimated change-points $\hat{\tau}_1, \ldots, \hat{\tau}_{\hat{v}}$

---

predicts a class label in {0, 4, 8, 12, 16, 22} (see Figure S6 in the online supplementary material) and 1 otherwise. The thresholding parameter $\gamma \in \mathbb{Z}^+$ is chosen to be 1/2. Figure 4 illustrates the result of multiple change-point detection in HASC data which provides evidence that the trained neural network can detect both the multiple change types and multiple change points.

7 Discussion

Reliable testing for change points and estimating their locations, especially in the presence of multiple change points, other heterogeneities or untidy data, is typically a difficult problem for the applied statistician: they need to understand what type of change is sought, be able to characterise it mathematically, find a satisfactory stochastic model for the data, formulate the appropriate statistic, and fine-tune its parameters. This makes for a long workflow, with scope for errors at its every stage.

In this paper, we showed how a carefully constructed statistical learning framework could automatically take over some of those tasks and perform many of them 'in one go' when provided with examples of labelled data. This turned the change-point detection problem into a supervised learning problem, and meant that the task of learning the appropriate test statistic and fine-tuning its parameters was left to the 'machine' rather than the human user.

The crucial question was that of choosing an appropriate statistical learning framework. The key factor behind our choice of neural networks was the discovery that the traditionally used likelihood-ratio-based change-point detection statistics could be viewed as simple neural networks, which (together with bounds on generalization errors beyond the training set) enabled us to formulate and prove the corresponding learning theory. However, there are a plethora of other excellent predictive frameworks, such as XGBoost, LightGBM or Random Forests (Breiman, 2001; Chen & Guestrin, 2016; Ke et al., 2017) and it would be of interest to establish whether and why they could or could not provide a viable alternative to neural nets here. Furthermore, if we view the neural network as emulating the likelihood-ratio test statistic, in that it will create test statistics for each possible location of a change and then amalgamate these into a single classifier, then we know that test statistics for nearby changes will often be similar. This suggests that imposing some smoothness on the weights of the neural network may be beneficial.
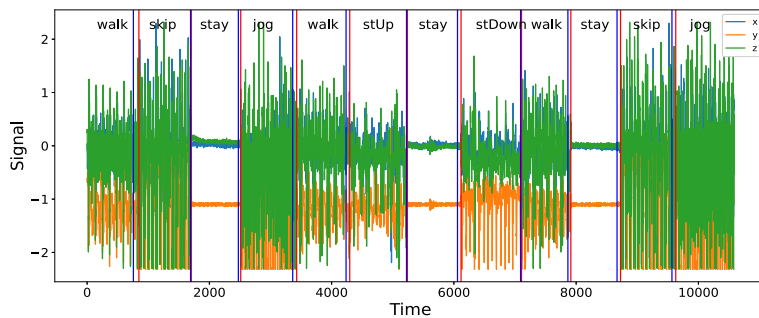
Figure 4. Change-point detection in HASC data. The red vertical lines represent the underlying change points, the blue vertical lines represent the estimated change points. More details on multiple change-point detection can be found in Section 3 of the online supplementary material.

A further challenge is to develop methods that can adapt easily to input data of different sizes, without having to train a different neural network for each input size. For changes in the structure of the mean of the data, it may be possible to use ideas from functional data analysis so that we pre-process the data, with some form of smoothing or imputation, to produce input data of the correct length.

If historical labelled examples of change points, perhaps provided by subject-matter experts (who are not necessarily statisticians) are not available, one question of interest is whether simulation can be used to obtain such labelled examples artificially, based on (say) a single dataset of interest. Such simulated examples would need to come in two flavours: one batch 'likely containing no change points' and the other containing some artificially induced ones. How to simulate reliably in this way is an important problem, which this paper does not solve. Indeed, we can envisage situations in which simulating in this way may be easier than solving the original unsupervised change-point problem involving the single dataset at hand, with the bulk of the difficulty left to the 'machine' at the learning stage when provided with the simulated data.

For situations where there is no historical data, but there are statistical models, one can obtain training data by simulation from the model. In this case, training a neural network to detect a change has similarities with likelihood-free inference methods in that it replaces analytic calculations associated with a model by the ability to simulate from the model. It is of interest whether ideas from that area of statistics can be used here.

The main focus of our work was on testing for a single offline change point, and we treated location estimation and extensions to multiple-change scenarios only superficially, via the heuristics of testing-based estimation in Section 6. Similar extensions can be made to the online setting once the neural network is trained, by retaining the final n observations in an online stream in memory and applying our change-point classifier sequentially. One question of interest is whether and how these heuristics can be made more rigorous: equipped with an offline classifier only, how can we translate the theoretical guarantee of this offline classifier to that of the corresponding location estimator or online detection procedure? In addition to this approach, how else can a neural network, however complex, be trained to estimate locations or detect change points sequentially? In our view, these questions merit further work.

Acknowledgements

Conflict of interest: We have no conflict of interest to disclose.

Funding

Data availability

The data underlying this article are available at http://hasc.jp/hc2011/index-en.html. The computer code and algorithm are available in Python Package: AutoCPD.

Supplementary material

Supplementary material is available online at *Journal of the Royal Statistical Society: Series B*.

References

Ahmadzadeh F. (2018). Change point detection with multivariate control charts by artificial neural network. *Journal of Advanced Manufacturing Technology*, 97(9), 3179–3190. https://doi.org/10.1007/s00170-009-2193-6

Baranowski R., Chen Y., & Fryzlewicz P. (2019). Narrowest-over-threshold detection of multiple change points and change-point-like features. *Journal of the Royal Statistical Society. Series B*, 81(3), 649–672. https://doi.org/10.1111/rssb.12322

Bartlett P. L., Harvey N., Liaw C., & Mehrabian A. (2019). Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63), 1–17. https://doi.org/10.48550/arXiv.1703.02930

Beaumont M. A. (2019). Approximate Bayesian computation. *Annual Review of Statistics and its Application*, 6(1), 379–403. https://doi.org/10.1146/statistics.2019.6.issue-1

Bos T., & Schmidt-Hieber J. (2022). Convergence rates of deep ReLU networks for multiclass classification. *Electronic Journal of Statistics*, 16(1), 2724–2773. https://doi.org/10.1214/22-EJS2011

Breiman L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324

Chang, W. -C., Li, C. -L., Yang, Y., & Póczos, B. (2019). Kernel change-point detection with auxiliary deep generative models. In *International Conference on Learning Representations, New Orleans, Louisiana, United States. May 6–May 9, 2019*.

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794). Association for Computing Machinery, New York, NY, United States.

Dehling H., Fried R., Garcia I., & Wendler M. (2015). Change-point detection under dependence based on two-sample U-statistics. In D. Dawson, R. Kulik, M. O. Haye, B. Szyszkowicz, & Y. Zhao (Eds.), *Asymptotic laws and methods in stochastics: A volume in honour of Miklós Csörgő* (pp. 195–220). Springer.

De Ryck T., De Vos M., & Bertrand A. (2021). Change point detection in time series data using autoencoders with a time-invariant representation. *IEEE Transactions on Signal Processing*, 69, 3513–3524. https://doi.org/10.1109/TSP.2021.3087031

Eichinger B., & Kirch C. (2018). A MOSUM procedure for the estimation of multiple random change points. *Bernoulli*, 24(1), 526–564. https://doi.org/10.3150/16-BEJ887

Fearnhead P., Maidstone R., & Letchford A. (2019). Detecting changes in slope with an l_0 penalty. *Journal of Computational and Graphical Statistics*, 28(2), 265–275. https://doi.org/10.1080/10618600.2018.1512868

Fearnhead P., & Rigaill G. (2020). Relating and comparing methods for detecting changes in mean. *Stat*, 9(1), 1–11. https://doi.org/10.1002/sta4.291

Fryzlewicz P. (2014). Wild binary segmentation for multiple change-point detection. *Annals of Statistics*, 42(6), 2243–2281. https://doi.org/10.1214/14-AOS1245

Fryzlewicz P. (2021). 'Robust narrowest significance pursuit: Inference for multiple change-points in the median', arXiv, arXiv:2109.02487, preprint: not peer reviewed.

Fryzlewicz P. (2023). Narrowest significance pursuit: Inference for multiple change-points in linear models. *Journal of the American Statistical Association*. https://doi.org/10.1080/01621459.2023.2211733

Gao Z., Shang Z., Du P., & Robertson J. L. (2019). Variance change point detection under a smoothly-changing mean trend with application to liver procurement. *Journal of the American Statistical Association*, 114(526), 773–781. https://doi.org/10.1080/01621459.2018.1442341

Gourieroux C., Monfort A., & Renault E. (1993). Indirect inference. *Journal of Applied Economics*, 8(S1), S85–S118. https://doi.org/10.1002/(ISSN)1099-1255

Gupta M., Wadhvani R., & Rasool A. (2022). Real-time change-point detection: A deep neural network-based adaptive approach for detecting changes in multivariate time series data. *Expert Systems with Applications*, 209, 1–16. https://doi.org/10.1016/j.eswa.2022.118260

Gutmann M. U., Dutta R., Kaski S., & Corander J. (2018). Likelihood-free inference via classification. *Statistics and Computing*, 28(2), 411–425. https://doi.org/10.1007/s11222-017-9738-6

Haynes K., Eckley I. A., & Fearnhead P. (2017). Computationally efficient changepoint detection for a range of penalties. *Journal of Computational and Graphical Statistics*, 26(1), 134–143. https://doi.org/10.1080/10618600.2015.1116445

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). IEEE.

Hocking T., Rigaill G., & Bourque G. (2015). PeakSeg: Constrained optimal segmentation and supervised penalty learning for peak detection in count data. In *International Conference on Machine Learning* (pp. 324–332). PMLR.

Huang, T. -J., Zhou, Q. -L., Ye, H. -J., & Zhan, D. -C. (2023). Change point detection via synthetic signals. In *8th Workshop on Advanced Analytics and Learning on Temporal Data* (pp. 25–35). AALTD 2023, Turin, Italy, September 18–22, 2023, Revised Selected Papers.

James B., James K. L., & Siegmund D. (1987). Tests for a change-point. *Biometrika*, 74(1), 71–83. https://doi.org/10.1093/biomet/74.1.71

Jandhyala V., Fotopoulos S., MacNeill I., & Liu P. (2013). Inference for single and multiple change-points in time series. *Journal of Time Series Analysis*, 34(4), 423–446. https://doi.org/10.1111/jtsa.12035

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T. -Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. In *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems* (pp. 3149–3157). Curran Associates Inc., 57 Morehouse Lane, Red Hook, NY, United States, December 2017.

Killick R., Fearnhead P., & Eckley I. A. (2012). Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, 107(500), 1590–1598. https://doi.org/10.1080/01621459.2012.737745

Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. In Y. Bengio & Y. LeCun (Eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7–9, 2015, Conference Track Proceedings* (pp. 1–15).

Lee J., Xie Y., & Cheng X. (2023). Training neural networks for sequential change-point detection. In *IEEE ICASSP 2023* (pp. 1–5). IEEE.

Li F., Tian Z., Xiao Y., & Chen Z. (2015). Variance change-point detection in panel data models. *Economics Letters*, 126, 140–143. https://doi.org/10.1016/j.econlet.2014.12.005

Liehrmann A., Rigaill G., & Hocking T. D. (2021). Increased peak detection accuracy in over-dispersed ChIP-seq data with supervised segmentation models. *BMC Bioinformatics*, 22(1), 1–18. https://doi.org/10.1186/s12859-021-04221-5

Londschien M., Bühlmann P., & Kovács S. (2022). 'Random forests for change point detection', arXiv, arXiv:2205.04997, preprint: not peer reviewed.

Mohri M., Rostamizadeh A., & Talwalkar A. (2012). *Foundations of machine learning*. Adaptive computation and machine learning series. MIT Press.

Oh K. J., Moon M. S., & Kim T. Y. (2005). Variance change point detection via artificial neural networks for data separation. *Neurocomputing*, 68, 239–250. https://doi.org/10.1016/j.neucom.2005.05.005

Paaß G., & Giesselbach S. (2023). *Foundation models for natural language processing: Pre-trained language models integrating media*. Artificial intelligence: Foundations, Theory, and Algorithms. Springer International Publishing.

Picard F., Robin S., Lavielle M., Vaisse C., & Daudin J.-J. (2005). A statistical approach for array CGH data analysis. *BMC Bioinformatics*, 6(1), 27. https://doi.org/10.1186/1471-2105-6-27

Reeves J., Chen J., Wang X. L., Lund R., & Lu Q. Q. (2007). A review and comparison of changepoint detection techniques for climate data. *Journal of Applied Meteorology and Climatology*, 46(6), 900–915. https://doi.org/10.1175/JAM2493.1

Ripley B. D. (1994). Neural networks and related methods for classification. *Journal of the Royal Statistical Society Series B*, 56(3), 409–456. https://doi.org/10.1111/j.2517-6161.1994.tb01990.x

Schmidt-Hieber J. (2020). Nonparametric regression using deep neural networks with ReLU activation function. *Annals of Statistics*, 48(4), 1875–1897. https://doi.org/10.1214/19-AOS1875

Shalev-Shwartz S., & Ben-David S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge University Press.

Truong C., Oudre L., & Vayatis N. (2020). Selective review of offline change point detection methods. *Signal Processing*, 167, 107299. https://doi.org/10.1016/j.sigpro.2019.107299

Wang T., & Samworth R. J. (2018). High dimensional change point estimation via sparse projection. *Journal of the Royal Statistical Society. Series B*, 80(1), 57–83. https://doi.org/10.1111/rssb.12243

Yamashita R., Nishio M., Do R. K. G., & Togashi K. (2018). Convolutional neural networks: An overview and application in radiology. *Insights Into Imaging*, 9(4), 611–629. https://doi.org/10.1007/s13244-018-0639-9

From denoising diffusions to denoising Markov models

Joe Benton[1] , Yuyang Shi[1], Valentin De Bortoli[2], George Deligiannidis[1] and Arnaud Doucet[1]

[1]Department of Statistics, University of Oxford, Oxford, UK
[2]ENS, Paris, France

Address for correspondence: Joe Benton, Department of Statistics, University of Oxford, 24-29 St Giles', Oxford OX1 3LB, UK. Email: benton@stats.ox.ac.uk

Read before The Royal Statistical Society at the Discussion Meeting on 'Probabilistic and statistical aspects of machine learning' held at the Society's 2023 annual conference in Harrogate on Wednesday, 6 September 2023, the President, Dr Andrew Garrett, in the Chair.

Abstract

Denoising diffusions are state-of-the-art generative models exhibiting remarkable empirical performance. They work by diffusing the data distribution into a Gaussian distribution and then learning to reverse this noising process to obtain synthetic datapoints. The denoising diffusion relies on approximations of the logarithmic derivatives of the noised data densities using score matching. Such models can also be used to perform approximate posterior simulation when one can only sample from the prior and likelihood. We propose a unifying framework generalizing this approach to a wide class of spaces and leading to an original extension of score matching. We illustrate the resulting models on various applications.

Keywords: denoising diffusions, generative models, posterior simulation, score matching, unifying framework

1 Introduction

Given a set of samples from an unknown distribution $p_{\text{data}}(\mathbf{x})$, generative modelling is the task of producing further synthetic samples coming from approximately the same distribution. Over the past decade, a variety of techniques have been developed to tackle this problem, including autoregressive models (Oord et al., 2016), generative adversarial networks (Goodfellow et al., 2014), variational autoencoders (Kingma & Welling, 2014), and normalizing flows (Rezende & Mohamed, 2015). These methods have had significant success in generating perceptually realistic samples from complex data distributions, such as text and image data (Brown et al., 2020; Dhariwal & Nichol, 2021). A major motivation for the development of generative models is that they can be easily extended for Bayesian inference. In a typical setting, we make an observation ξ^* based on underlying datapoint $\mathbf{x}$, for example a category label or partial observation of $\mathbf{x}$, and want to sample from the posterior distribution $p_{\text{data}}(\mathbf{x} \mid \xi^*)$. We achieve this by learning a conditional generative model for $\mathbf{x}$ given any observation ξ based on samples from $p_{\text{data}}(\mathbf{x}, \xi)$. This approach is particularly useful in high-dimensional scenarios where traditional sampling methods, such as Markov chain Monte Carlo (MCMC) methods or approximate Bayesian computation (ABC), are typically infeasible.

Recently, denoising diffusion models (Ho et al., 2020; Sohl-Dickstein et al., 2015; Song et al., 2021) have emerged as effective generative models for high-dimensional data. They work by incrementally adding noise to the data to transform the data distribution into an easy-to-sample reference distribution, and then learning to invert the noising process, which is achieved using score matching

(Hyvärinen, 2005). Their use for inference has recently seen an explosion of applications, including text-to-speech generation (Popov et al., 2021), image inpainting and super-resolution (Saharia et al., 2022; Song et al., 2021), and protein structure modelling (Trippe et al., 2023).

Most of the current methodology, theory, and applications of denoising diffusion models are for diffusion processes on $\mathbb{R}^d$. However, many distributions of interest are defined on different spaces. Recently, De Bortoli et al. (2022) and Huang et al. (2022) have extended continuous-time methods and the analogy with score matching from $\mathbb{R}^d$ to general Riemannian manifolds in order to model data with strong geometric prior. Several diffusion methods have also been developed for discrete data, such as text, music, or graph structures (Austin et al., 2021; Campbell et al., 2022; Hoogeboom et al., 2021; Sun et al., 2023). Here though, the relationships to score matching, as well as between these various methods and the Euclidean diffusion case, are less clear. All these recent extensions have been somewhat ad hoc, with training objectives needing to be re-derived for each new application.

The main contribution of this paper is to provide a unifying framework for such models, which we call *denoising Markov models*, or DMMs. We demonstrate how to construct and train a DMM for data in any state space satisfying mild regularity conditions. This yields a principled procedure for using these models for unconditional generation and inference on a wider class of spaces than previously considered. Additionally this general framework leads to a principled extension of score matching to general spaces. Finally, we demonstrate the application of our framework on examples in continuous space, discrete space, for Riemmanian manifolds and on the simplex.

2 Background

A denoising diffusion model is a generative model consisting of two stochastic processes. The *fixed* noising process takes a data point $\mathbf{x}_0$ drawn from a data distribution $q_0 := p_{\text{data}}$ on state space $\mathcal{X}$ and maps it stochastically to some $\mathbf{x}_T \in \mathcal{X}$. The *learned* generative process takes $\mathbf{x}_T \in \mathcal{X}$ drawn according to some initial distribution p_0 on $\mathcal{X}$ and maps it back stochastically to some $\mathbf{x}_0 \in \mathcal{X}$. Throughout, we denote the marginals of the noising and generative processes by $q_t(\mathbf{x})$ and $p_t(\mathbf{x})$, respectively, for $t \in [0, T]$.

The basic idea is to pick a noising process so that $(q_t)_{t \geq 0}$ converges to some easy-to-sample-from distribution q_{ref}, which we then take to be p_0. We learn a generative process which approximates the time-reversal of the noising process. Then, we can generate approximate samples from q_0 by sampling $\mathbf{x}_T \sim p_0$ and running the dynamics of the reverse process to produce a sample $\mathbf{x}_0 \sim p_T$, which should be close to q_0.

2.1 Continuous-time denoising diffusion models on $\mathbb{R}^d$

The framework for continuous-time diffusion models on $\mathbb{R}^d$ was first set out by Song et al. (2021). The noising process $(Y_t)_{t \in [0,T]}$ evolves according to the stochastic differential equation (SDE)

$$\mathrm{d}Y_t = b(Y_t, t)\, \mathrm{d}t + \mathrm{d}B_t, \qquad Y_0 = \mathbf{x}_0 \sim p_{\text{data}}, \tag{1}$$

for some chosen function $b : \mathbb{R}^d \times [0, T] \to \mathbb{R}^d$, and standard Brownian motion B. With this set-up, the time-reversed process $X_t = Y_{T-t}$ can be simulated by initializing $X_0 = \mathbf{x}_T \sim q_T$ and running the SDE

$$\mathrm{d}X_t = \{ -b(X_t, T - t) + \nabla_{\mathbf{x}} \log q_{T-t}(X_t) \}\, \mathrm{d}t + \mathrm{d}\hat{B}_t, \tag{2}$$

where $q_t(\mathbf{x}_t)$ denotes the marginals of the forward process and $\hat{B}$ is another standard Brownian motion (Anderson, 1982). We typically choose our forward process to be an Ornstein–Uhlenbeck process, i.e. $b(\mathbf{x}, t) = -\mathbf{x}/2$, for which $q_T \approx q_{\text{ref}} := \mathcal{N}(0, I_d)$, the standard Gaussian distribution on $\mathbb{R}^d$, for large T.

To simulate the reverse process, we must approximate $\nabla_{\mathbf{x}} \log q_t(\mathbf{x})$. We do this by fixing a parametric family of functions $s_\theta(\mathbf{x}, t)$, and then choosing the parameters θ to minimize the *denoising score matching* objective

$$\mathcal{I}_{\text{DSM}}(\theta) = \frac{1}{2} \int_0^T \mathbb{E}_{q_{0,t}(\mathbf{x}_0, \mathbf{x}_t)} \left[\| \nabla_{\mathbf{x}} \log q_{t|0}(\mathbf{x}_t | \mathbf{x}_0) - s_\theta(\mathbf{x}_t, t) \|^2 \right] \mathrm{d}t, \tag{3}$$

where $q_{0,t}(\mathbf{x}_0, \mathbf{x}_t)$ and $q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)$ denote the joint and conditional distributions of the SDE (1). The conditional is available in closed-form for the Ornstein–Uhlenbeck process. This is sensible since $\mathcal{I}_{\mathrm{DSM}}$ is minimized when $s_\theta(\mathbf{x}, t) = \nabla_\mathbf{x} \log q_t(\mathbf{x})$ for almost all $x \in \mathcal{X}$ and $t \in [0, T]$ (Song et al., 2021). If our score estimate were exact and $p_0 = q_T$, then we would have $p_t = q_{T-t}$ for all $t \in [0, T]$. In practice, we use a neural network to parameterize $s_\theta(\mathbf{x}, t)$ and use stochastic gradient descent to minimize $\mathcal{I}_{\mathrm{DSM}}(\theta)$.

Once we have a score estimate $s_\theta(\mathbf{x}, t)$, we compute approximate samples from the reverse process by running the approximate reverse process

$$\mathrm{d}X_t = \{ - b(X_t, T - t) + s_\theta(X_t, T - t)\} \, \mathrm{d}t + \mathrm{d}\hat{B}_t \tag{4}$$

starting in $X_0 \sim p_0$ and setting $\mathbf{x}_0 = X_T$. In practice, we use suitable numerical integrators to simulate the approximate reverse process.

Alternatively, the objective $\mathcal{I}_{\mathrm{DSM}}$ can be derived from a lower bound on the model log-likelihood (also known as an evidence lower bound or ELBO) for $q_T(x)$, either using Girsanov's theorem and the chain rule for Kullback–Leibler divergences (Song et al., 2021), or by combining the Fokker–Planck equation and Feynman–Kac formula with Girsanov's theorem (Huang et al., 2021).

2.2 Diffusion models for inference

Denoising diffusions can also be used to sample approximately from a posterior $p_{\mathrm{data}}(\mathbf{x} \mid \boldsymbol{\xi}^*)$ when we only have access to samples from the joint distribution $p_{\mathrm{data}}(\mathbf{x}, \boldsymbol{\xi})$; see, e.g. Song et al. (2021). We first draw a sample $(\mathbf{x}_0, \boldsymbol{\xi}_0) \sim p_{\mathrm{data}}$, set $Y_0 = \mathbf{x}_0$ and let $(Y_t)_{t\in[0,T]}$ evolve according to equation (1). If we condition on $\boldsymbol{\xi}_0$, then the process Y has marginals $q_t(\mathbf{x}_t \mid \boldsymbol{\xi}_0) = \int q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0) p_{\mathrm{data}}(\mathbf{x}_0 \mid \boldsymbol{\xi}_0) \, \mathrm{d}\mathbf{x}_0$, where $q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)$ is the transition kernel of the forward diffusion in equation (1). So, the time-reversed process $X_t = Y_{T-t}$ conditioned on $\boldsymbol{\xi}_0$ can be simulated by initializing $X_0 \sim q_T(\cdot \mid \boldsymbol{\xi}_0)$ and running the SDE

$$\mathrm{d}X_t = \{ - b(X_t, T - t) + \nabla_\mathbf{x} \log q_{T-t}(X_t \mid \boldsymbol{\xi}_0)\} \, \mathrm{d}t + \mathrm{d}\hat{B}_t. \tag{5}$$

If we have $q_T(\cdot \mid \boldsymbol{\xi}) \approx q_{\mathrm{ref}}$ for all $\boldsymbol{\xi}$ and an approximation $s_\theta(\mathbf{x}, \boldsymbol{\xi}, t)$ to $\nabla_\mathbf{x} \log q_t(\mathbf{x} \mid \boldsymbol{\xi})$, we can obtain approximate samples from $q_0(\cdot \mid \boldsymbol{\xi}^*) = p_{\mathrm{data}}(\cdot \mid \boldsymbol{\xi}^*)$ for any given $\boldsymbol{\xi}^*$ by initializing $X_0 \sim p_0 := q_{\mathrm{ref}}$, simulating the reverse dynamics in equation (5) with $\nabla_\mathbf{x} \log q_{T-t}(X_t \mid \boldsymbol{\xi}_0)$ replaced by $s_\theta(X_t, \boldsymbol{\xi}^*, T - t)$, and setting $\mathbf{x}_0 = X_T$. To learn $s_\theta(\mathbf{x}, \boldsymbol{\xi}, t)$, we minimize

$$\mathcal{I}_{\mathrm{DSM}}(\theta) = \frac{1}{2} \int_0^T \mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_t, \boldsymbol{\xi}_0)} \big[\| \nabla_\mathbf{x} \log q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0) - s_\theta(\mathbf{x}_t, \boldsymbol{\xi}_0, t) \|^2 \big] \, \mathrm{d}t,$$

where we denote $q(\mathbf{x}_0, \mathbf{x}_t, \boldsymbol{\xi}_0) = p_{\mathrm{data}}(\mathbf{x}_0, \boldsymbol{\xi}_0) q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)$. This objective is minimized when $s_\theta(\mathbf{x}, \boldsymbol{\xi}, t) = \nabla_\mathbf{x} \log q_t(\mathbf{x} \mid \boldsymbol{\xi})$ for almost all $x \in \mathcal{X}$ and $t \in [0, T]$ (Song et al., 2021).

2.3 Score matching

The objective $\mathcal{I}_{\mathrm{DSM}}$ defined in equation (3) can also be interpreted as a score matching objective. Score matching was introduced as a method for fitting unnormalized probability distributions defined on $\mathbb{R}^d$ by Hyvärinen (2005). It approximates a distribution $q_0(\mathbf{x})$ with a distribution of the form $p(\mathbf{x}; \theta) = q(\mathbf{x}; \theta)/Z(\theta)$ by minimizing

$$\mathcal{J}(\theta) = \frac{1}{2} \mathbb{E}_{q_0(\mathbf{x})} \big[\| \nabla_\mathbf{x} \log q_0(\mathbf{x}) - \nabla_\mathbf{x} \log q(\mathbf{x}; \theta) \|^2 \big],$$

known as an *explicit score matching* loss. This objective is intractable since it depends on $\nabla_\mathbf{x} \log q_0(\mathbf{x})$, but there are methods for rewriting it in an equivalent tractable form, including implicit and denoising score matching (Hyvärinen, 2005; Vincent, 2011). Equation (3), which corresponds to denoising score matching, can also be written in explicit, implicit, or sliced score matching form (Huang et al., 2021).
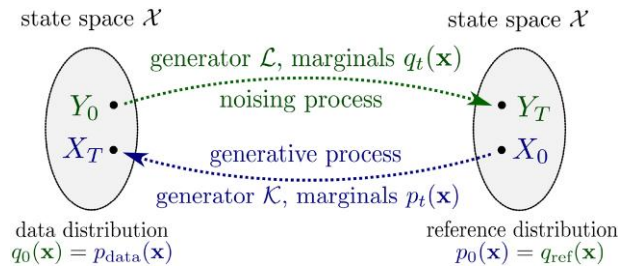
Figure 1. Diagram of notation.

3 A general framework for DMMs

In this section, we set out a general framework for DMMs. First, we explain how to construct a DMM on an arbitrary state space with a forward noising process Y and backward generative process X. Second, we derive an expression for the model likelihood in terms of an expectation over an auxiliary process Z, defined in terms of X and running forward in time. Third, we derive an ELBO by using Girsanov's theorem to relate the expectation over Z to one over Y. Finally, we show how this ELBO can be used to get a tractable training objective. Our argument follows a similar structure to Huang et al. (2021), but we work in terms of generic Markov generators, rather than specific operators corresponding to diffusions on $\mathbb{R}^d$, and so require generalizations of the stochastic process results therein. For simplicity, we present the framework for unconditional generation and then explain how to adapt it for inference.

3.1 Notation and set-up

Our data is assumed to be distributed according to p_{data} on a state space $\mathcal{X}$. We assume only that $\mathcal{X}$ comes with some reference measure v, with respect to which all probability densities will be defined, and satisfies some regularity conditions given in online supplementary Appendix B.1. This includes $\mathbb{R}^d$, discrete spaces and Riemannian manifolds (with or without boundary).

Our DMM consists of a noising process $(Y_t)_{t \in [0,T]}$ and a generative process $(X_t)_{t \in [0,T]}$, which are Markov processes. We consider Y fixed and learn X to approximate the reverse of Y. Initially, we must fix a class of processes to which X and Y belong and within which we will optimize X. The particular class and parameterization we choose will necessarily depend on $\mathcal{X}$, but a typical choice for $\mathcal{X} = \mathbb{R}^d$ would be a diffusion (see Example 1), while a typical choice when $\mathcal{X}$ is a finite discrete space may be a continuous-time Markov chain (CTMC) (see Example 2). Our notation is depicted in Figure 1.

As X and Y are not necessarily time-homogeneous, it is helpful to define the extended processes $\overline{X}$ and $\overline{Y}$ by for example setting $X_t = X_T$ for $t \geq T$ and letting $\overline{X} = (X_t, t)_{t \geq 0}$. Then $\overline{X}, \overline{Y}$ are time-homogeneous Markov chains on the extended space $\mathcal{S} := \mathcal{X} \times [0, \infty)$.

In general, it is most convenient to define X and Y via the generators of $\overline{X}$ and $\overline{Y}$, which we denote by $\mathcal{K}$ and $\mathcal{L}$, respectively. Informally, the generator of a Markov process $\overline{W}$ with state space $\mathcal{S}$ is an operator $\mathcal{A}$ which acts on a subset $\mathcal{D}(\mathcal{A})$ of the space of functions $f : \mathcal{S} \to \mathbb{R}$ and satisfies $\mathcal{A}f = \lim_{s \to 0} (P_s f - f)/s$, where $(P_s)_{s \geq 0}$ is the transition semi-group associated to $\overline{W}$ and $P_s f(x) = \mathbb{E}\big[f(X_s) \mid X_0 = x\big]$. For a more formal definition, see online supplementary Appendix A.1.

We denote the time marginals of the processes X, Y by $p_t(\mathbf{x})$, $q_t(\mathbf{x})$, respectively. We make some smoothness assumptions on p, in online supplementary Appendix B.2, and assume that $\mathcal{K}, \mathcal{L}$ satisfy some regularity conditions, in online supplementary Appendix B.3. Our assumptions hold for standard models in the literature (Euclidean diffusions, CTMCs and manifold diffusions; see online supplementary Appendix F), plus some that are not covered previously, such as degenerate diffusions. For infinite-dimensional spaces, the assumptions of online supplementary Appendix B.1 may fail and more care is needed.

One consequence of our assumptions is that the operator $\mathcal{K}$ decomposes as $\mathcal{K} = \partial_t + \hat{\mathcal{K}}$, where $\hat{\mathcal{K}}$ operates only on the spatial variables of a function f. We can therefore view $\hat{\mathcal{K}}$ as an operator on

functions from $\mathcal{X}$, rather than on functions from $\mathcal{S}$, and we denote by $\hat{\mathcal{K}}^*$ the adjoint of $\hat{\mathcal{K}}$ acting on functions on $\mathcal{X}$ (see online supplementary Appendix A.2).

> **Example 1** (Euclidean diffusion). If X and Y are diffusions on $\mathbb{R}^d$ given by the SDEs $\mathrm{d}X_t = \mu(X_t, t)\,\mathrm{d}t + \mathrm{d}\hat{B}_t$ and $\mathrm{d}Y_t = b(Y_t, t)\,\mathrm{d}t + \mathrm{d}B_t$, where B and $\hat{B}$ are Brownian motions, then the corresponding generators are $\mathcal{K} = \partial_t + \mu \cdot \nabla + \frac{1}{2}\Delta$ and $\mathcal{L} = \partial_t + b \cdot \nabla + \frac{1}{2}\Delta$, where $\Delta = \sum_{i=1}^d \frac{\partial^2}{\partial x_i^2}$ denotes the Laplacian. We then have $\hat{\mathcal{K}}^* = -\mu \cdot \nabla - (\nabla \cdot \mu) + \frac{1}{2}\Delta$ using integration by parts.

> **Example 2** (Discrete-space CTMC). If X and Y are CTMCs, then $\mathcal{K} = \partial_t + A$ and $\mathcal{L} = \partial_t + B$, where A and B are the time-dependent generator matrices of X and Y. In this case, $\hat{\mathcal{K}}^* = A^T$, the transpose of A.

3.2 An expression for the model likelihood

We now derive an expression for the model likelihood $p_T(\mathbf{x})$. First, under our assumptions, a generalized form of the Fokker–Planck equation, stated precisely in online supplementary Appendix C, implies that $\partial_t p = \hat{\mathcal{K}}^* p$ for v-almost every $\mathbf{x} \in \mathcal{X}$. Typically, the adjoint operator $\hat{\mathcal{K}}^*$ resembles the generator of another process in the same class as X and Y. We formalize this idea by making the following assumption.

> **Assumption 1** Let $v(\mathbf{x}, t) = p_{T-t}(\mathbf{x})$. Then we can write the equation $\partial_t p = \hat{\mathcal{K}}^* p$ in the form $\mathcal{M}v + cv = 0$ for some function $c : \mathcal{S} \to \mathbb{R}$, where $\mathcal{M}$ is the generator of another auxiliary Feller process $\overline{Z} = (Z_t, t)_{t \geq 0}$ on $\mathcal{S}$.

> **Example 3** (Euclidean diffusion). For Euclidean diffusions, the Fokker–Planck equation can be written as $\partial_t v = \mu \cdot \nabla v + (\nabla \cdot \mu)v - \frac{1}{2}\Delta v$. Assumption 1 is satisfied with $c = -(\nabla \cdot \mu)$ and $\mathcal{M} = \partial_t - \mu \cdot \nabla + \frac{1}{2}\Delta$, noting that $\mathcal{M}$ is the generator of the diffusion process Z defined by $\mathrm{d}Z_t = -\mu(Z_t, T - t)\,\mathrm{d}t + \mathrm{d}B_t'$, where B' is a Brownian motion.

> **Example 4** (Discrete-space CTMC). In the CTMC case, if $c_\mathbf{x} = \sum_{\mathbf{y} \in \mathcal{X}} A_{\mathbf{yx}}$ and $D_{\mathbf{xy}} = A_{\mathbf{yx}} - c_\mathbf{x}\mathbb{1}_{\mathbf{x=y}}$, then $\mathcal{M} = \partial_t + D$ is the generator of a CTMC and Assumption 1 is satisfied. Here c has a natural interpretation as a 'discrete divergence'.

In general, we make two smoothness assumptions on c and v, given in online supplementary Appendix B.4.

Given the Fokker–Planck equation and Assumption 1, we apply a generalized form of the Feynman–Kac Theorem (see online supplementary Appendix C) to $\overline{Z}$ and v to get the following expression for the model likelihood, which generalizes that of Huang et al. (2021):

$$p_T(\mathbf{x}) = v(\mathbf{x}, 0) = \mathbb{E}\Big[p_0(Z_T)\exp\Big\{\int_0^T c(Z_s, s)\,\mathrm{d}s\Big\} \mid Z_0 = \mathbf{x}\Big]. \tag{6}$$

This gives an expression in terms of an expectation over the auxiliary process Z. We next make this tractable by converting it into an expectation over Y.

3.3 Deriving a tractable lower bound on the model log-likelihood

We would like to train our model by finding a reverse process X which maximizes the likelihood in equation (6). Unfortunately this expression is intractable, but we can find a tractable lower bound for $\log p_T(\mathbf{x})$ which can then be used as a surrogate objective.

By taking logarithms in equation (6) and applying Jensen's inequality, we get

$$\log p_T(\mathbf{x}) \geq \mathbb{E}_{\mathbb{Q}}\left[\log \frac{\mathrm{d}\mathbb{P}}{\mathrm{d}\mathbb{Q}} + \log p_0(Y_T) + \int_0^T c(Y_s, s)\,\mathrm{d}s \mid Y_0 = \mathbf{x}\right] =: \mathcal{E}^\infty, \tag{7}$$

where $\mathbb{P}$ and $\mathbb{Q}$ are the path measures of the processes $\overline{Z}$ and $\overline{Y}$, respectively, and $\frac{\mathrm{d}\mathbb{P}}{\mathrm{d}\mathbb{Q}}$ denotes the Radon–Nikodym derivative.

To write $\mathcal{E}^\infty$ in a tractable form, we need to evaluate $\log \frac{\mathrm{d}\mathbb{P}}{\mathrm{d}\mathbb{Q}}$, which we do using a generalization of Girsanov's theorem. To apply this result, we require that the generators of the auxiliary process and the noising process are related in the following way.

> **Assumption 2**   There is a bounded measurable function $\beta: \mathcal{S} \to (0, \infty)$ such that $\beta^{-1}\mathcal{M}f = \mathcal{L}(\beta^{-1}f) - f\mathcal{L}(\beta^{-1})$ for all $f: S \to \mathbb{R}$ such that $f \in \mathcal{D}(\mathcal{M})$ and $\beta^{-1}f \in \mathcal{D}(\mathcal{L})$.

Since $\mathcal{M}$ is defined in terms of $\mathcal{K}$, we think of Assumption 2 as forcing a particular parameterization of the generative process in terms of β. In general, not every generative process in the same class as $\mathcal{L}$ will have such a parameterization. However, the true time-reversal of $\mathcal{L}$ can always be parameterized in this way with $\beta(\mathbf{x}, t) = p_t(\mathbf{x})$, so this parameterization is sufficient to capture the optimal generative process. In addition, the objective in Theorem 1 below can often be interpreted and used for a much broader set of generative processes than those which satisfy Assumption 2.

Under Assumption 2, along with a further technical assumption given in online supplementary Appendix B.5, we may apply a generalized form of Girsanov's Theorem (see online supplementary Appendix C), and take $\alpha = \beta^{-1}$ in Theorem 6 and Dynkin's formula (see online supplementary Appendix A.1) to get

$$\log \frac{\mathrm{d}\mathbb{P}}{\mathrm{d}\mathbb{Q}} = \int_0^T \left\{ -\mathcal{L}\log\beta(Y_s, s) - \beta(Y_s, s)\mathcal{L}(\beta^{-1})(Y_s, s) \right\}\mathrm{d}s + \mathbb{Q}\text{-martingale}.$$

In addition, we get that $c = \beta\mathcal{L}(\beta^{-1}) - v^{-1}\beta\mathcal{L}(\beta^{-1}v)$ by combining Assumption 2 with $f = v$ and Assumption 1. This allows us to rewrite the ELBO from equation (7) as

$$\mathcal{E}^\infty = \mathbb{E}_{\mathbb{Q}}\left[\log p_0(Y_T) - \int_0^T \left\{ \frac{\mathcal{L}(\beta^{-1}v)}{\beta^{-1}v} + \mathcal{L}\log\beta \right\}\mathrm{d}s \,\middle|\, Y_0 = \mathbf{x}\right].$$

The final step required to get a tractable expression for $\mathcal{E}^\infty$ is to remove the function v from this expression. For this, we use the following lemma (see online supplementary Appendix D).

> **Lemma 1**   Let the generator $\mathcal{L}$ and the functions β and c be as above. Then, we have $v^{-1}\beta\mathcal{L}(\beta^{-1}v) + \mathcal{L}\log\beta = \beta^{-1}\hat{\mathcal{L}}^*\beta + \hat{\mathcal{L}}\log\beta$.

> **Theorem 1**   For DMMs as in Sections 3.1–3.3, the log-likelihood is lower bounded by
>
> $$\mathcal{E}^\infty = \mathbb{E}_{\mathbb{Q}}[\log p_0(Y_T) \mid Y_0 = \mathbf{x}] - \int_0^T \mathbb{E}_{\mathbb{Q}}\left[\frac{\hat{\mathcal{L}}^*\beta}{\beta} + \hat{\mathcal{L}}\log\beta \,\middle|\, Y_0 = \mathbf{x}\right]\mathrm{d}s. \tag{8}$$

This result extends the corresponding expression for $\mathbb{R}^d$ in Huang et al. (2021). We see the ELBO consists of a term representing the log-likelihood under the reference distribution and an implicit score matching term arising from the change in measure.

3.4 Finding suitable training objectives

Based on Theorem 1, we fit our generative model by maximizing the expectation of $\mathcal{E}^\infty$ with respect to p_{data}. This is equivalent to minimizing the objective

$$\mathcal{I}_{\text{ISM}}(\beta) = \int_0^T \mathbb{E}_{q_t(\mathbf{x}_t)} \left[\frac{\hat{\mathcal{L}}^* \beta(\mathbf{x}_t, t)}{\beta(\mathbf{x}_t, t)} + \hat{\mathcal{L}} \log \beta(\mathbf{x}_t, t) \right] dt, \tag{9}$$

which we call the *implicit score matching* objective, since it can be interpreted as an extension of implicit score matching from $\mathbb{R}^d$ (see Section 4 below for more intuition).

Since q_t and $\hat{\mathcal{L}}$ are determined by the noising process, which is known and assumed easy to sample from, $\mathcal{I}_{\text{ISM}}(\beta)$ and its gradient with respect to β can be estimated in an unbiased fashion. Since β parameterizes $\mathcal{M}$ via Assumption 2, and thus $\mathcal{K}$ through Assumption 1, minimizing $\mathcal{I}_{\text{ISM}}(\beta)$ over β is equivalent to learning the generative process.

We also have an equivalent *denoising score matching objective* (see online supplementary Appendix E),

$$\mathcal{I}_{\text{DSM}}(\beta) = \int_0^T \mathbb{E}_{q_{0,t}(\mathbf{x}_0, \mathbf{x}_t)} \left[\frac{\mathcal{L}(q_{\cdot|0}(\cdot \mid \mathbf{x}_0)/\beta(\cdot, \cdot))(\mathbf{x}_t, t)}{q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)/\beta(\mathbf{x}_t, t)} - \mathcal{L} \log (q_{\cdot|0}(\cdot \mid \mathbf{x}_0)/\beta(\cdot, \cdot))(\mathbf{x}_t, t) \right] dt. \tag{10}$$

Both objectives are minimized when $\beta(\mathbf{x}, t) \propto q_t(\mathbf{x})$, as shown in Proposition 1. $\mathcal{I}_{\text{DSM}}(\beta)$ can be interpreted as quantifying the difference between $q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)$ and $\beta(\mathbf{x}_t, t)$ via the score matching operator $\Phi(f) = f^{-1} \mathcal{L} f - \mathcal{L} \log f$ introduced in Section 4 below. These objectives also generalize the following previously studied instances of diffusion models. For all derivations and remarks on the choice of parameterization, see online supplementary Appendix F.

> **Example 5** (Euclidean diffusion). In the setting of Example 1, Assumption 2 reduces to $\nabla \log \beta = b + \mu$, and we have $f^{-1} \mathcal{L} f - \mathcal{L} \log f = \frac{1}{2} \| \nabla \log f \|^2$. If we substitute $s_\theta(\mathbf{x}_t, t) = \nabla \log \beta(\mathbf{x}_t, t)$, $\mathcal{I}_{\text{DSM}}(\beta)$ defined in equation (10) reduces to equation (3) and the reverse process is parameterized as in equation (4). We thus recover the results of Song et al. (2021) and Huang et al. (2021).

> **Example 6** (Discrete-space CTMC). In the setting of Example 2, Assumption 2 reduces to $A_{\mathbf{yx}} = \frac{\beta(\mathbf{x}, t)}{\beta(\mathbf{y}, t)} B_{\mathbf{xy}}$ for all $\mathbf{x} \neq \mathbf{y}$. We may rewrite $\mathcal{I}_{\text{ISM}}$ in terms of A to recover the objective of Campbell et al. (2022),
>
> $$\mathcal{I}_{\text{ISM}}(A) = \int_0^T \mathbb{E}_{q_t(\mathbf{x}_t)} \left[-A_{\mathbf{x}_t \mathbf{x}_t} - \sum_{\mathbf{y} \neq \mathbf{x}_t} B_{\mathbf{x}_t \mathbf{y}} \log A_{\mathbf{yx}_t} \right] dt + const.$$

> **Example 7** (Riemannian manifolds). If $\mathcal{X}$ is a Riemannian manifold and we take $\mathcal{K} = \partial_t + \mu \cdot \nabla + \frac{1}{2} \Delta$, $\mathcal{L} = \partial_t + b \cdot \nabla + \frac{1}{2} \Delta$, where Δ is the Laplace–Beltrami operator associated to $\mathcal{X}$, and perform the reparameterization $s_\theta(\mathbf{x}_t, t) = \nabla \log \beta(\mathbf{x}_t, t)$, then we recover the framework for training diffusion models on Riemannian manifolds given in De Bortoli et al. (2022) and Huang et al. (2022).

3.5 Inference

To use DMMs for inference, we follow a similar procedure to Section 2.2. To noise a sample $(\mathbf{x}_0, \xi_0) \sim p_{\text{data}}$, we set $Y_0 = \mathbf{x}_0$ and let Y evolve according to $\mathcal{L}$. To generate $\mathbf{x}_0$ conditioned on an observation ξ^*, we use a generative process X^{ξ^*} conditioned on ξ^*. We parameterize X^{ξ^*} in terms of a function $\beta(\mathbf{x}_t, \xi^*, t)$ which now takes ξ^* as an input.

We aim to learn X^{ξ^*} to approximate the time-reversal of Y conditioned on ξ^*. The following extension of Theorem 1 (proved in online supplementary Appendix D) gives us a way to do this.

Theorem 2   With the above set-up, minimizing the objective

$$\mathcal{I}_{\text{DSM}}(\beta) = \int_0^T \mathbb{E}_{q(\mathbf{x}_0, \mathbf{x}_t, \xi_0)}$$
$$\left[\frac{\mathcal{L}(q_{\cdot|0}(\cdot \mid \mathbf{x}_0)/\beta(\cdot, \xi_0, \cdot))(\mathbf{x}_t, t)}{q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)/\beta(\mathbf{x}_t, \xi_0, t)} - \mathcal{L} \log \left(q_{\cdot|0}(\cdot \mid \mathbf{x}_0)/\beta(\cdot, \xi_0, \cdot) \right)(\mathbf{x}_t, t) \right] dt$$

is equivalent to maximizing a lower bound on the expected model log-likelihood.

Theorem 2 suggests that we may train conditional DMMs by maximizing the objective $\mathcal{I}_{\text{DSM}}(\beta)$ (or the equivalent $\mathcal{I}_{\text{ISM}}(\beta)$ objective). Since $q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)$ is known, we may do this by calculating an empirical estimate for $\mathcal{I}_{\text{DSM}}(\beta)$ based on samples $(\mathbf{x}_0, \xi_0)$ drawn from p_{data} and minimizing over β. Then, we generate samples from $p_{\text{data}}(\mathbf{x}_0 \mid \xi^*)$ by initializing $X_0^{\xi^*} \sim p_0$, simulating the reverse process with generator $\mathcal{K}$ parameterized by $\beta = \beta(\cdot, \xi^*, \cdot)$, and setting $\mathbf{x}_0 = X_T^{\xi^*}$.

4 Score matching on general state spaces

When X and Y are Euclidean diffusions, the objective $\mathcal{I}_{\text{DSM}}(\beta)$ in equation (10) becomes the score matching objective in equation (3). Similarly, the objective $\mathcal{I}_{\text{ISM}}(\beta)$ from equation (9) reduces to the implicit score matching objective introduced by Hyvärinen (2005). This suggests we can view equations (9) and (10) as generalizations of score matching objectives to arbitrary state spaces.

Given state space $\mathcal{X}$ on which we have a Markov process generator $\mathcal{L}$ and an unknown distribution $q_0(\mathbf{x})$ we wish to approximate, the corresponding *generalized implicit score matching* method learns an approximation $\varphi(\mathbf{x})$ to $q_0(\mathbf{x})$ by minimizing

$$\mathcal{J}_{\text{ISM}}(\varphi) = \mathbb{E}_{q_0(\mathbf{x})} \left[\frac{\hat{\mathcal{L}}^* \varphi(\mathbf{x})}{\varphi(\mathbf{x})} + \hat{\mathcal{L}} \log \varphi(\mathbf{x}) \right].$$

We can show that $\mathcal{J}_{\text{ISM}}$ is equivalent to the *generalized explicit score matching objective*

$$\mathcal{J}_{\text{ESM}}(\varphi) = \mathbb{E}_{q_0(\mathbf{x})} \left[\frac{\mathcal{L}(q_0/\varphi)(\mathbf{x})}{(q_0(\mathbf{x})/\varphi(\mathbf{x}))} - \mathcal{L} \log (q_0/\varphi)(\mathbf{x}) \right].$$

In addition, we define the corresponding *generalized denoising score matching* method, which learns an approximation $\varphi_\tau(\mathbf{x}_\tau)$ to the noised distribution $q_\tau(\mathbf{x}_\tau)$, formed by sampling $\mathbf{x}_0 \sim q_0(\cdot)$ and $\mathbf{x}_\tau \sim q_{\tau|0}(\cdot \mid \mathbf{x}_0)$, where $q_{\tau|0}$ is the transition probability associated to $\mathcal{L}$ run for time τ. It does this by minimizing the objective

$$\mathcal{J}_{\text{DSM}}(\varphi_\tau) = \mathbb{E}_{q_{0,\tau}(\mathbf{x}_0, \mathbf{x}_\tau)} \left[\frac{\mathcal{L}(q_{\tau|0}(\cdot \mid \mathbf{x}_0)/\varphi_\tau(\cdot))(\mathbf{x}_\tau)}{q_{\tau|0}(\mathbf{x}_\tau \mid \mathbf{x}_0)/\varphi_\tau(\mathbf{x}_\tau)} - \mathcal{L} \log (q_{\tau|0}(\cdot \mid \mathbf{x}_0)/\varphi_\tau(\cdot))(\mathbf{x}_\tau) \right].$$

$\mathcal{J}_{\text{DSM}}$ is equivalent to both $\mathcal{J}_{\text{ISM}}$ and $\mathcal{J}_{\text{ESM}}$ when used to learn the smoothed distribution $q_\tau(\mathbf{x}_\tau)$ (see online supplementary Appendix E). All three objectives extend the corresponding score matching objectives introduced for $\mathbb{R}^d$ by Hyvärinen (2005) and Vincent (2011). They also coincide with the extension of score matching for Riemannian manifolds of Mardia et al. (2016).

To illustrate further intuitions behind our objective functions, we define the *score matching operator* $\Phi(f) = f^{-1} \mathcal{L} f - \mathcal{L} \log f$. Note that the time component of Φ cancels, so we can view it as an operator on $\mathcal{X}$. With this notation, the generalized explicit score matching objective becomes $\mathcal{J}_{\text{ESM}}(\varphi) = \mathbb{E}_{q_0(\mathbf{x})} \left[\Phi(q_0/\varphi)(\mathbf{x}) \right]$. For Euclidean diffusions, $\Phi(f) = \frac{1}{2} \|\nabla \log f\|^2$ (see Example 5). In the general case, we view $\Phi(f)$ as measuring the magnitude of a logarithmic gradient of f. We interpret the objectives $\mathcal{J}_{\text{DSM}}$ and $\mathcal{J}_{\text{ESM}}$ as trying to fit φ to q_0 by minimizing this logarithmic gradient of the ratio q_0/φ.

Proposition 1   Let Y be a Feller process with semi-group operators $(Q_t)_{t\geq 0}$, generator $\mathcal{L}$ and associated score matching operator Φ. Then:

(a) $\Phi(f) \geq 0$ for all f in the domain of Φ, with equality if f is constant;
(b) for any probability measures π_1, π_2 on $\mathcal{X}$ and $t \geq 0$,

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathrm{KL}(\pi_1 Q_t \| \pi_2 Q_t) = -\mathbb{E}_{\pi_1 Q_t}\left[\Phi\left(\frac{\mathrm{d}(\pi_1 Q_t)}{\mathrm{d}(\pi_2 Q_t)}\right)\right],$$

where $\mathrm{KL}(\pi_1 Q_t \| \pi_2 Q_t)$ denotes the Kullback–Leibler divergence between $\pi_1 Q_t$ and $\pi_2 Q_t$.

Proposition 1(a) shows that Φ is always non-negative, so $\mathcal{J}_{\mathrm{ESM}}$ is minimized if $\varphi(\mathbf{x}) \propto q_0(\mathbf{x})$. Thus, minimizing any of our generalized score matching objectives should typically correspond to learning an approximation to q_0. Note though that if Q_t is not ergodic and π_1, π_2 are different invariant distributions of Q_t then Proposition 1(b) implies that $\Phi(\mathrm{d}\pi_1/\mathrm{d}\pi_2) = 0$ π_1-a.e., even though $\mathrm{d}\pi_1/\mathrm{d}\pi_2$ is not constant. This suggests that generalized score matching may fail if the noising process is not ergodic. Proposition 1(b) was proved for score matching on $\mathbb{R}^d$ by Lyu (2009). It suggests we can interpret score matching as finding an approximation φ which minimizes the decrease in KL divergence between q_0 and φ caused by adding an infinitesimal amount of noise to both according to $\mathcal{L}$.

Our generalized score matching methods give a principled way to extend score matching to fit unnormalized probability distributions on arbitrary spaces. Other extensions of score matching have been explored, including to arbitrary sub-domains of $\mathbb{R}^d$ (Yu et al., 2022), ratio matching (Hyvärinen, 2007) and marginalization with generalized score matching (Lyu, 2009). However, these methods lack the generality of our framework and do not respect the intuition coming from $\mathbb{R}^d$ that Proposition 1(b) should hold. There are also many other density estimation methods that seek to learn ratios of density functions, including noise-contrastive estimation, which also approximates score matching under certain conditions (Gutmann & Hirayama, 2011).

5 Relationship to discrete time models

Denoising diffusion models were originally introduced in discrete time by Sohl-Dickstein et al. (2015). In this setting, the noising and generative processes are Markov chains $\mathbf{x}_{0:T} = (\mathbf{x}_{t_k})_{k=0}^N$ observed at a sequence of times $0 = t_0 < t_1 < \cdots < t_N = T$, with fixed forwards transition kernel $\tilde{q}(\mathbf{x}_{t_k} \mid \mathbf{x}_{t_{k-1}})$ and learned backwards kernel $\tilde{p}_\theta(\mathbf{x}_{t_{k-1}} \mid \mathbf{x}_{t_k})$. To fit discrete time diffusion models, Sohl-Dickstein et al. (2015) minimize the following Kullback–Leibler divergence with respect to θ:

$$\mathrm{KL}(\tilde{q}(\mathbf{x}_{0:T}) \| \tilde{p}_\theta(\mathbf{x}_{0:T})) = \sum_{k=1}^N \mathbb{E}_{\tilde{q}(\mathbf{x}_{t_{k-1}}, \mathbf{x}_{t_k})}\left[\log \frac{\tilde{q}(\mathbf{x}_{t_k} \mid \mathbf{x}_{t_{k-1}})}{\tilde{p}_\theta(\mathbf{x}_{t_{k-1}} \mid \mathbf{x}_{t_k})}\right] + const. \tag{11}$$

Given any DMM with generators $\mathcal{K}$, $\mathcal{L}$ and marginals p_t, q_t as in Section 3, we define its *natural discretization* to be the discrete-time model with $\tilde{q}(\mathbf{x}_{t_k} \mid \mathbf{x}_{t_{k-1}}) = q_{t_k|t_{k-1}}(\mathbf{x}_{t_k}|\mathbf{x}_{t_{k-1}})$ and $\tilde{p}_\theta(\mathbf{x}_{t_{k-1}} \mid \mathbf{x}_{t_k}) = p_{T-t_{k-1}|T-t_k}(\mathbf{x}_{t_{k-1}}|\mathbf{x}_{t_k})$. Then, the Kullback–Leibler divergence (11) for the natural discretization can be viewed as a first-order approximation to $\mathcal{I}_{\mathrm{ISM}}$ for the continuous-time model.

Lemma 2   Suppose X, Y are fixed generative and noising processes with marginals p, q as in Section 3, and suppose that they are related as in Assumptions 1 and 2 for some sufficiently regular function β. Then for any $0 < s < t < T$ with $\gamma = t - s$,

$$\gamma\,\mathbb{E}_{q_s(\mathbf{x}_s)}\left[\frac{\hat{\mathcal{L}}^*\beta}{\beta} + \hat{\mathcal{L}}\log\beta\right] = \mathbb{E}_{q_{s,t}(\mathbf{x}_s, \mathbf{x}_t)}\left[\log\frac{q_{t|s}(\mathbf{x}_t \mid \mathbf{x}_s)}{p_{T-s|T-t}(\mathbf{x}_s \mid \mathbf{x}_t)}\right] + o(\gamma).$$

Applying this lemma on each interval $[t_k, t_{k+1}]$, we get the following theorem.

Theorem 3   For any DMM, the objective (11) for its natural discretization is equivalent to the natural discretization of $\mathcal{I}_{\mathrm{ISM}}$ to first order in $\bar{\gamma} = \max_{k=0,\dots,N-1} |t_{k+1} - t_k|$.
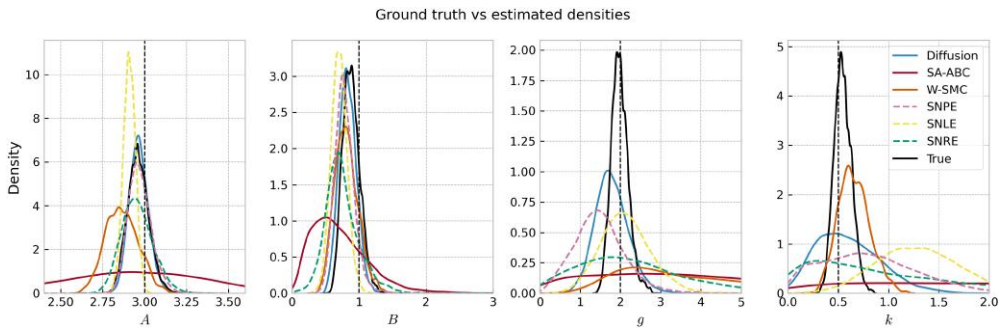
Figure 2. Posterior kernel density estimates of samples generated using our DMM, SA-ABC, W-SMC, SNLE, SNPE, and SNRE for the *g*-and-*k* distribution, with $\mathbf{x}_{\text{true}} = (3, 1, 2, 0.5)$ and $N = 250$. DMM = denoising Markov model; SA-ABC = semi-automatic approximate Bayesian computation; W-SMC = Wasserstein sequential Monte Carlo; SNLE = sequential neural likelihood estimation; SNPE = sequential neural posterior estimation; SNRE = sequential neural ratio estimation.

This theorem generalizes to arbitrary state spaces a result of Ho et al. (2020), which demonstrated the equivalence of minimizing (11) and the score matching objective for Euclidean state spaces. For the proofs of Lemma 2 and Theorem 3, see online supplementary Appendix H.

Lemma 2 also implies a general equivalence between one-step denoising autoencoders and score matching. Vincent (2011) discussed this equivalence for autoencoders using Gaussian noise in $\mathbb{R}^d$, but our methods allow us to extend this correspondence to arbitrary state spaces and noising processes. For more details, see online supplementary Appendix I.

6 Experiments

We now present experiments demonstrating DMMs on several tasks and data spaces, for unconditional generation and conditional simulation. All details are in online supplementary Appendix J.

6.1 Inference on $\mathbb{R}^d$ using diffusion processes

First, we use diffusion processes in $\mathbb{R}^d$ to perform approximate Bayesian inference for real-valued parameters. We consider $p_{\text{data}}(\boldsymbol{\xi} \mid \mathbf{x}) = \prod_{i=1}^{N} p_{\text{data}}(\xi_i \mid \mathbf{x})$, where $p_{\text{data}}(\xi_i \mid \mathbf{x})$ is the *g*-and-*k* distribution with parameters $\mathbf{x} = (A, B, g, k)$ and $d = 4$, and we let $p_{\text{data}}(\mathbf{x})$ be uniform on $[0, 10]^4$. The *g*-and-*k* distribution is a four-parameter distribution in which A, B, g, k control the location, scale, skewness, and kurtosis, respectively.

We fix our noising process to be an Ornstein–Uhlenbeck process, and parameterize our reverse process as in Example 5, with $s_\theta(\mathbf{x}, \boldsymbol{\xi}, t)$ being given by a fully connected neural network. To train the model, we sample $(\mathbf{x}_0, \boldsymbol{\xi}_0) \sim p_{\text{data}}$ and minimize the denoising score matching objective from Section 3.5 via stochastic gradient descent on θ.

To test our model, we first consider the case where there are a true set of underlying parameters $\mathbf{x}_{\text{true}} = (3, 1, 2, 0.5)$. We generate an observation $\boldsymbol{\xi}_0 \sim p_{\text{data}}(\boldsymbol{\xi}_0 \mid \mathbf{x}_{\text{true}})$ with $N = 250$, sample from the approximate posterior using our DMM and plot the result in Figure 2. We compare our method with the semi-automatic ABC (SA-ABC) (Nunes & Prangle, 2015) and Wasserstein sequential Monte Carlo (W-SMC) (Bernton et al., 2019) methodologies, as well as sequential neural posterior, likelihood, and ratio estimation approaches (SNPE, SNLE, and SNRE) (see, e.g. Lueckmann et al., 2021). We see in Figure 2 that the DMM achieves more accurate posterior estimation for all parameters, except the kurtosis parameter *k* for which W-SMC is more accurate. Among the other neural network-based approaches, SNPE appears most competitive on this task, but is less accurate than the DMM especially for parameters *g* and *k*. Additional experimental results comparing DMMs to other simulation-based inference methods can be found in Sharrock et al. (2022) and Geffner et al. (2023).

Next, we demonstrate that our model can perform inference for a range of observation values $\boldsymbol{\xi}^*$ simultaneously. We generate a series of 512 parameter values $\mathbf{x}_0$ drawn from $p_{\text{data}}(\mathbf{x}_0)$ and draw an
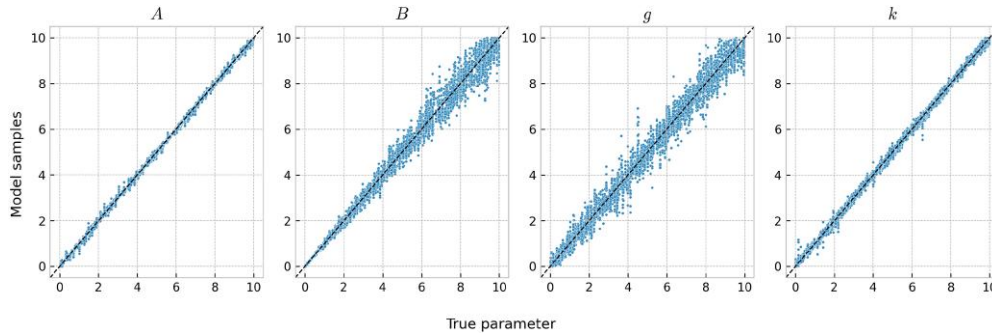
Figure 3. Comparison of posterior samples $\mathbf{x}_0'$ from our DMM approximation to $p_{\text{data}}(\cdot \mid \xi_0)$ and the true parameter value $\mathbf{x}_0$ for a range of $\mathbf{x}_0$ in the prior distribution, with $N = 10{,}000$. DMM = denoising Markov model.

observation ξ_0 from $p_{\text{data}}(\xi_0 \mid \mathbf{x}_0)$ with $N = 10,000$ for each $\mathbf{x}_0$. Then, we generate eight samples $\mathbf{x}_0'$ from our approximation to the posterior $p_{\text{data}}(\mathbf{x}_0 \mid \xi_0)$ for each ξ_0. We plot each component of the pairs $(\mathbf{x}_0, \mathbf{x}_0')$ in Figure 3. We see our model is able to infer the original parameters across a range of parameter values.

6.2 Image inpainting and super-resolution using discrete-space CTMCs

Second, we demonstrate that our framework is applicable for large-scale Bayesian inverse problems, such as super-resolution and inpainting for images. For these problems, the prior $p_{\text{data}}(\mathbf{x})$ is the distribution of images. Most ABC techniques such as SA-ABC and W-SMC are not applicable as they require an analytical expression for this prior, whereas DMMs do not rely on such an expression.

We consider performing image inpainting for MNIST digit images, where each image $\mathbf{x}_0$ has 28×28 pixels with values in $\{0, \ldots, 255\}$, and the observed incomplete image ξ_0 has the middle 14×14 pixels missing. Since our state space $\mathcal{X} = \{0, \ldots, 255\}^{28 \times 28}$ is discrete, we use the set-up of Example 2 and let the generator of our noising process factor over pixel dimensions. We use the denoising parameterization of the reverse process (see online supplementary Appendix F.2) and train by minimizing the form of the objective in Example 6.

To test our model, we plot the reconstructed image samples for a number of digits in Figure 4. We observe that the samples we obtain are consistent with conditioning and appear to be realistic, but also display diversity in the shape of the strokes. In online supplementary Appendix J.2, we also compare our method to a continuous state-space approach.

In addition, we train a conditional discrete-space DMM to perform super-resolution on ImageNet images to demonstrate that this method provides perceptually high-quality samples even in very high-dimensional scenarios. For details, see online supplementary Appendix J.3.

6.3 Modelling distributions on *SO*(3) using manifold diffusions

Third, we demonstrate that DMMs can approximate distributions on manifolds using two tasks on $SO(3)$. Since $SO(3)$ is a Lie group and so a Riemannian manifold, we use the framework from Example 7. As our noising process, we use Brownian motion with generator $\mathcal{L} = \partial_t + \frac{1}{2}\Delta$. We can explicitly calculate the transition kernels $q_{t|0}(\mathbf{x}_t \mid \mathbf{x}_0)$ for this process, allowing us to use the denoising score matching objective. We parameterize this objective in terms of a neural network approximation $s_\theta(\mathbf{x}, t)$ of the score. This is in contrast to De Bortoli et al. (2022), in which the explicit transition kernels are not used for sampling the forward process or in the loss function, both of which require further approximations.

First, we check that our DMM can learn simple mixtures of wrapped normal distributions $p_{\text{data}}(\mathbf{x}) = \frac{1}{M}\sum_{m=1}^{M} \mathcal{N}^{\text{W}}(\mathbf{x} \mid \mu_m, \sigma_m^2)$, where $\mathcal{N}^{\text{W}}(\mathbf{x} \mid \mu_m, \sigma_m^2)$ is the wrapped normal distribution on $SO(3)$ with expectation μ_m and variance σ_m^2 (De Bortoli et al., 2022). We plot samples from our resulting DMM in Figure 5. We see that our model provides a good fit to $p_{\text{data}}(\mathbf{x})$, covering all
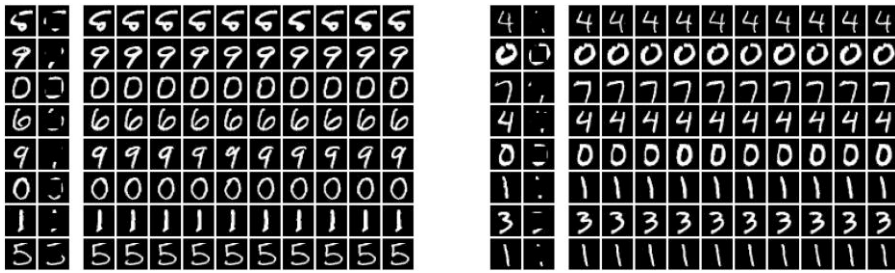
Figure 4. Samples from the MNIST inpainting task. The first column in each set plots the ground truth images, and the second column has the centre 14×14 pixels missing.
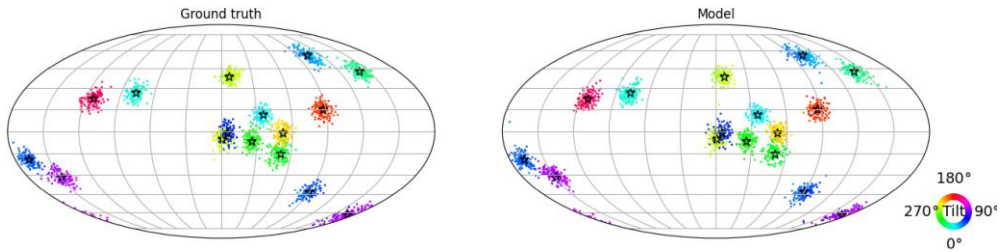

Figure 5. Samples from the ground truth and our DMM approximation to the mixture of wrapped normal distributions. Each sample is denoted by a point, whose position represents the axis of rotation and whose hue represents the angle of rotation. Stars denote the true cluster means. DMM = denoising Markov model.
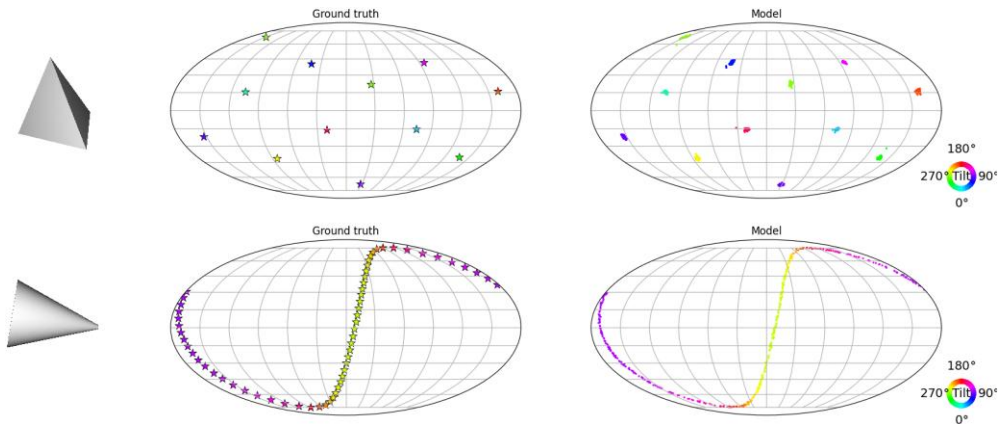

Figure 6. Samples from the ground truth (plotted as stars, middle) and our pose estimation DMM (right) conditioned on 2D views of two shapes (left). The axis of rotation and rotation angle are represented by position and hue, respectively. DMM = denoising Markov model.

modes. In online supplementary Appendix J.5, we provide additional results and show that we can also sample from the class conditional density $p_{\text{data}}(\mathbf{x} \mid m)$.

Second, we consider a more realistic pose estimation task on the SYMSOL dataset, which requires predicting the 3D orientation of various symmetric 3D solids based on 2D views (Murphy et al., 2021). Due to the rotational symmetries, a key challenge is to predict all possible poses when only one possibility is presented in training. We use a conditional DMM where ξ is the 2D image view. Figure 6 shows two sets of samples from our model conditioned on 2D images of two different solids. We see that our model learns to sample from the ground truth accurately and
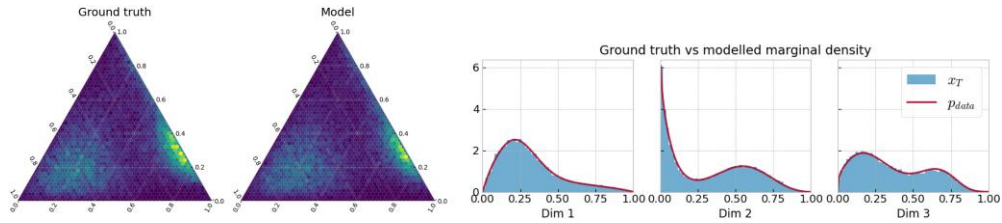
Figure 7. Histograms of samples from our simplex DMM and the ground truth mixture of Dirichlet distributions for dimension $N = 3$, plotted over the whole space as a ternary plot (left) and over the marginals per dimension (right). DMM = denoising Markov model.

infer the full set of rotational symmetries for different views ξ. For further experimental details and plots, see online supplementary Appendix J.6.

6.4 Approximation of distributions over measures using Wright–Fisher diffusions

Finally, we present an example of learning to approximate a distribution over measures on a finite state space $E = \{1, \ldots, N\}$. In this case $\mathcal{X} = \mathcal{P}(E)$, the space of measures on E. This is of particular interest in compositional data analysis (Greenacre, 2021). Elements of $\mathcal{X}$ can be parameterized by tuples of real numbers $\mathbf{p} = (p_1, \ldots, p_N) \in [0, 1]^N$ such that $\sum_{i=1}^N p_i = 1$. We could approximate the data distribution using a diffusion model on $\mathbb{R}^N$, but such a model would not reflect the fact that our distribution should be supported on a submanifold, the simplex. Using the standard set-up for manifold diffusions as in Example 7 would not respect the boundary of the simplex. Other methods have been presented in the literature, but they rely on either reflected diffusions (Lou & Ermon, 2023) or on projections of the simplex (Richemond et al., 2022).

We therefore use Wright–Fisher diffusions, a process used in population genetics to model the evolution of allele frequencies, as our class of generative processes. A Wright–Fisher process has generator $\mathcal{L} = \partial_t + \frac{1}{2} \sum_{i,j=1}^N p_i(\delta_{ij} - p_j) \frac{\partial^2}{\partial p_i \partial p_j} + \sum_{i,j=1}^N q_{ij} p_i \frac{\partial}{\partial p_j}$, where $(q_{ij})_{i,j=1,\ldots,N}$ is some matrix such that $\sum_{j=1}^N q_{ij} = 0$ for each $i = 1, \ldots, N$. The process takes values in the space of measures on E, and so respects the structure of our data distribution (Ethier & Griffiths, 1993). For specific choices of q_{ij}, the process converges to a known invariant distribution and we can calculate the implicit score matching loss. For details of the theoretical set-up, see online supplementary Appendix F.4.

We evaluate the proposed method by modelling $p_{\text{data}}(\mathbf{x}) = \frac{1}{M} \sum_{m=1}^M \text{Dirichlet}(\alpha_m)$, a mixture of Dirichlet distributions with parameters $\alpha_m \in \mathbb{R}^N$, for various values of N. Figure 7 shows two visualizations of samples drawn from our DMM compared to ground truth samples in dimension $N = 3$. Our model is able to accurately approximate $p_{\text{data}}(\mathbf{x})$. For further evaluations and experimental details, see online supplementary Appendix J.7.

7 Discussion

We have provided here a general framework which allows us to extend denoising diffusion models to general state spaces. The resulting DMMs can be trained with principled objectives and used for inference, generalizing along the way score matching ideas. Their applicability and performance have been demonstrated on a range of problems. From a methodological point of view, the proposed framework is general enough to accommodate, for example, general noising processes, mixed continuous/discrete processes and some infinite-dimensional settings with finite representations (though our assumptions on the state space (see online supplementary Appendix B.1) may fail to hold in the infinite-dimensional setting so more care is required).

However, we still lack a proper theoretical understanding of these models. Under realistic assumptions on the data distribution, De Bortoli (2023) and Chen et al. (2023) show that diffusion models on $\mathbb{R}^d$ can in theory learn essentially any distribution given a good enough score approximation and infinite data. However, finite sample guarantees are currently absent. Moreover, p_{data} is typically an empirical measure as we only have access to a finite set of datapoints, so q_t is a mixture of Gaussians for an Ornstein–Uhlenbeck noising diffusion and its score $\nabla \log q_t$ is thus

available. If we were simulating samples using the exact time reversal of this diffusion, we would simply recover the empirical distribution. It is because we are approximating the time-reversal and in particular using an approximation of the scores that we are able to obtain novel samples. It is not yet clear why the approximation of the score using neural networks appears to provide perceptually realistic samples for many applications.

The effectiveness of such methods for inference, even in scenarios where standard MCMC or ABC techniques are not applicable (Geffner et al., 2023; Sharrock et al., 2022), may also be considered surprising. One perspective on the training process is that it involves the model constructing its own summary statistics that allow it to perform inference effectively on the training observations. It is not yet well understood why the summary statistics the model learns appear empirically effective, or what sorts of summary statistics our training procedure biases the model towards.

Overall, this contribution shows how the range of existing models relate to each other and may help applying DMMs in practice to a large variety of problems. However, our understanding of such models is still incomplete and deserves further attention.

Acknowledgments

Funding

Supplementary material

Supplementary material is available online at *Journal of the Royal Statistical Society: Series B.*

References

Anderson B. D. O. (1982). Reverse-time diffusion equation models. *Stochastic Processes and Their Applications*, *12*(3), 313–326. https://doi.org/10.1016/0304-4149(82)90051-5

Austin J., Johnson D. D., Ho J., Tarlow D., & van den Berg R. (2021). Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, *34*, 17981–17993. https://doi.org/10.48550/arXiv.2107.03006

Bernton E., Jacob P. E., Gerber M., & Robert C. P. (2019). Approximate Bayesian computation with the Wasserstein distance. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *81*(2), 235–269. https://doi.org/10.1111/rssb.12312

Brown T., Mann B., Ryder N., Subbiah M., Kaplan J. D., Dhariwal P., Neelakantan A., Shyam P., Sastry G., & Askell A. (2020). Language models are few-shot learners. *Advances in Neural Information Processing Systems*, *33*, 1877–1901. https://doi.org/10.48550/arXiv.2005.14165

Campbell A., Benton J., Bortoli V. De, Rainforth T., Deligiannidis G., & Doucet A. (2022). A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*, *35*, 28266–28279. https://doi.org/10.48550/arXiv.2205.14987

Chen S., Chewi S., Li J., Li Y., Salim A., & Zhang A. R. (2023). Sampling is as easy as learning the score: Theory for diffusion models with minimal data assumptions. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2209.11215

De Bortoli V. (2023). Convergence of denoising diffusion models under the manifold hypothesis. *Transactions on Machine Learning Research*. https://doi.org/10.48550/arXiv.2208.05314

De Bortoli V., Mathieu E., Hutchinson M., Thornton J., Teh Y. W., & Doucet A. (2022). Riemannian score-based generative modeling. *Advances in Neural Information Processing Systems*, *35*, 2406–2422. https://doi.org/10.48550/arXiv.2202.02763

Dhariwal P., & Nichol A. (2021). Diffusion models beat GANs on image synthesis. *Advances in Neural Information Processing Systems*, *34*, 8780–8794. https://doi.org/10.48550/arXiv.2105.05233

Ethier S. N., & Griffiths R. C. (1993). The transition function of a Fleming-Viot process. *The Annals of Probability*, *21*(3), 1571–1590. https://doi.org/10.1214/aop/1176989131

Geffner T., Papamakarios G., & Mnih A. (2023). 'Compositional score modeling for simulation-based inference', arXiv, arXiv:2209.14249, preprint: not peer reviewed.

Goodfellow I. J., Pouget-Abadie J., Mirza M., Xu B., Warde-Farley D., Ozair S., Courville A., & Bengio Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems*, *27*. https://doi.org/10.48550/arXiv.1406.2661

Greenacre M. (2021). Compositional data analysis. *Annual Review of Statistics and its Application*, *8*(1), 271–299. https://doi.org/10.1146/statistics.2021.8.issue-1

Gutmann M. U., & Hirayama J.-I. (2011). Bregman divergence as general framework to estimate unnormalized statistical models. *Uncertainty in Artificial Intelligence*. https://doi.org/10.48550/arXiv.1202.3727

Ho J., Jain A., & Abbeel P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, *33*, 6840–6851. https://doi.org/10.48550/arXiv.2006.11239

Hoogeboom E., Nielsen D., Jaini P., Forré P., & Welling M. (2021). Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in Neural Information Processing Systems*, *34*, 12454–12465. https://doi.org/10.48550/arXiv.2102.05379

Huang C.-W., Aghajohari M., Bose A. J., Panangaden P., & Courville A. (2022). Riemannian diffusion models. *Advances in Neural Information Processing Systems*, *35*, 2750–2761. https://doi.org/10.48550/arXiv.2208.07949

Huang C.-W., Lim J. H., & Courville A. (2021). A variational perspective on diffusion-based generative models and score matching. *Advances in Neural Information Processing Systems*, *34*, 22863–22876. https://doi.org/10.48550/arXiv.2106.02808

Hyvärinen A. (2005). Estimation of non-normalized statistical models by score matching. *Journal of Machine Learning Research*, *6*, 695–709.

Hyvärinen A. (2007). Some extensions of score matching. *Computational Statistics and Data Analysis*, *51*(5), 2499–2512. https://doi.org/10.1016/j.csda.2006.09.003

Kingma D. P., & Welling M. (2014). Auto-encoding variational Bayes. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.1312.6114

Lou A., & Ermon S. (2023). 'Reflected diffusion models', arXiv, arXiv:2304.04740, preprint: not peer reviewed.

Lueckmann J.-M., Boelts J., Greenberg D. S., Gonçalves P. J., & Macke J. H. (2021). Benchmarking simulation-based inference. *Artificial Intelligence and Statistics*. https://doi.org/10.48550/arXiv.2101.04653

Lyu S. (2009). Interpretation and generalization of score matching. *Uncertainty in Artificial Intelligence*. https://doi.org/10.48550/arXiv.1205.2629

Mardia K. V., Kent J. T., & Laha A. K. (2016). 'Score matching estimators for directional distributions', arXiv, arXiv:1604.08470, preprint: not peer reviewed.

Murphy K. A., Esteves C., Jampani V., Ramalingam S., & Makadia A. (2021). Implicit-PDF: Non-parametric representation of probability distributions on the rotation manifold. In *ICML* (pp. 7882–7893).

Nunes M. A., & Prangle D. (2015). abctools: An R package for tuning approximate Bayesian computation analyses. *The R Journal*, *7*(2), 189–205. https://doi.org/10.32614/RJ-2015-030

Oord A. v. d., Dieleman S., Zen H., Simonyan K., Vinyals O., Graves A., Kalchbrenner N., Senior A., & Kavukcuoglu K. (2016). 'WaveNet: A generative model for raw audio', arXiv, arXiv:1609.03499, preprint: not peer reviewed.

Popov V., Vovk I., Gogoryan V., Sadekova T., & Kudinov M. (2021). Grad-TTS: A diffusion probabilistic model for text-to-speech. *International Conference on Machine Learning*. https://doi.org/10.48550/arXiv.2105.06337

Rezende D. J., & Mohamed S. (2015). Variational inference with normalizing flows. *International Conference on Machine Learning*. https://doi.org/10.48550/arXiv.1505.05770

Richemond P. H., Dieleman S., & Doucet A. (2022). 'Categorical SDEs with simplex diffusion', arXiv, arXiv:2210.14784, preprint: not peer reviewed.

Saharia C., Ho J., Chan W., Salimans T., Fleet D. J., & Norouzi M. (2022). Image super-resolution via iterative refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *45*(4), 4713–4726. https://doi.org/10.48550/arXiv.2104.07636

Sharrock L., Simons J., Liu S., & Beaumont M. (2022). 'Sequential neural score estimation: Likelihood-free inference with conditional score based diffusion models', arXiv, arXiv:2210.04872, preprint: not peer reviewed.

Sohl-Dickstein J., Weiss E. A., Maheswaranathan N., & Ganguli S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. *International Conference on Machine Learning*. https://doi.org/10.48550/arXiv.1503.03585

Song Y., Durkan C., Murray I., & Ermon S. (2021). Maximum likelihood training of score-based diffusion models. *Advances in Neural Information Processing Systems*, *34*, 1415–1428. https://doi.org/10.48550/arXiv.2101.09258

Song Y., Sohl-Dickstein J., Kingma D. P., Kumar A., Ermon S., & Poole B. (2021). Score-based generative modeling through stochastic differential equations. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2011.13456

Sun H., Yu L., Dai B., Schuurmans D., & Dai H. (2023). Score-based continuous-time discrete diffusion models. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2211.16750

Trippe B. L., Yim J., Tischer D., Baker D., Broderick T., Barzilay R., & Jaakkola T. (2023). Diffusion probabilistic modeling of protein backbones in 3D for the motif-scaffolding problem. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2206.04119

Vincent P. (2011). A connection between score matching and denoising autoencoders. *Neural Computation*, *23*(7), 1661–1674. https://doi.org/10.1162/NECO_a_00142

Yu S., Drton M., & Shojaie A. (2022). Generalized score matching for general domains. *Information and Inference: A Journal of the IMA*, *11*(2), 739–780. https://doi.org/10.1093/imaiai/iaaa041

Discussion Paper Contribution

Proposer of the vote of thanks and contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Darren Wilkinson ⓘ

Department of Mathematical Sciences, Durham University, Durham, UK
Address for correspondence: Darren Wilkinson, Department of Mathematical Sciences, Durham University, South Road, Durham DH1 3LE, UK. Email: darren.j.wilkinson@durham.ac.uk

Computational statistics and machine learning (ML) are closely related, and there are many opportunities for cross-fertilization of ideas between the two fields. Both can benefit from greater interaction, and the two papers being discussed here highlight some ways that this can happen.

1 Automatic change-point detection in time series via deep learning

The main focus of this paper is offline detection of a single change-point using labelled training data. Interest is in the automatic generation of new offline detection methods using neural networks whilst providing statistical guarantees of method performance. Theory is developed for a class of multi-layer perceptrons (MLPs) that directly generalize existing cumulative sum-based methods.

The theory developed in the paper applies to a MLP with ReLU activation, and this basic model is amenable to analysis. However, the theory only requires a single layer, and the examples all use MLPs of constant layer width, which is rarely seen in practice. Can the authors provide practical advice on choosing network depth and layer widths sensibly and safely? In particular, are there practical issues relating to the width condition, $m_r m_{r+1} = \mathcal{O}(n \log n)$? The theoretical bounds suggest the need for a lot of training data, but empirically it seems that these may be overly conservative. Do the authors have any insight into this apparent mismatch? Neural networks often work better with scaled data, and min–max scaling is used for the examples in the paper, but is this safe in the presence of heavy-tailed noise?

For the application based on activity data, a more sophisticated neural network architecture is adopted for which the theoretical results provided do not directly apply. What hope is there of extending the theory to such models, and in the absence of this, what practical advice can be given? It is mentioned in the paper that in the absence of labelled data, but in the presence of a full data generating process, a simulator can be used to train the network. This is *simulation-based inference* (SBI) (Cranmer et al., 2020), and it is worth establishing the connection with this literature, where the use of neural networks has become a standard practice in recent years.

2 From denoising diffusions to denoising Markov models

Denoising Markov models (DMMs) are deep generative models for simulating (conditional) samples from a data distribution. Huge training data sets are required, but these can be replaced by a

data generating process in the context of SBI. Typical applications are to very high dimensional data such as images, but the methods can also be used for sampling Bayesian posterior distributions. These models are very expensive both to train and sample (even relative to other deep generative models), but are considered state-of-the-art for certain problems.

The paper provides a unifying framework for a broad class of models of the denoising diffusion form with fairly arbitrary state spaces. The emphasis is on continuous time, but the connection with discrete time formulations is clearly articulated. Conditional simulation is also covered and briefly discussed. The approach is to work with continuous time Markov processes on a general state space, and to formulate the (de)noising process in terms of the generator of the Markov process. The resulting optimization targets are shown to generalize several different special cases that have appeared in the literature for particular state spaces.

Details of how to generate samples are missing from the main paper, but are very important in practice. The examples described in the online supplementary material seem to use approximate first-order methods based on a regular time grid, but this is probably not optimal. A benefit of formulating the models in continuous time is the possibility of using higher-order methods with adaptive time steps. At least one of the examples used a time-rescaling—might adaptive time-stepping reduce the need for this, or is that a separate issue? It is sometimes convenient to have a deterministic generation mechanism using a probability flow differential equation (Song et al., 2021). Is such an approach covered by the general DMM framework presented here? What about Schrödinger bridge (Shi et al., 2022) approaches?

Everything depends on using a 'good' neural network architecture for the denoising process, but can anything general be said about how to choose the architecture for a given problem? Do we understand the kinds of problems for which DMMs work well? Why are not these models more widely used for SBI? Given the magnitude of the computational machinery dedicated to the problem, the g-and-k example was not especially compelling (see, e.g. Figure 8 in the online supplementary material), despite being a fairly standard low-dimensional Bayesian inference problem. Could issues be diagnosed in the absence of ground truth, and could the model be tuned to improve performance if desired? Are there examples in the literature of DMMs being used for problems with a mixed discrete and continuous state space?

3 Summary

These two papers illustrate different aspects of the interaction between statistics and ML. From the perspective of academic statistics, we are likely to see increasing use of modern ML methods in statistical methodology. It is likely to become difficult to draw a clear line between computational statistics and ML, but this comes with challenges, since the language and culture of the two communities remain quite distinct. Programming languages also illustrate potential issues: Python is the language typically used for ML, with tensor frameworks such as TensorFlow (used for Paper 1), JAX (used for Paper 2), and Torch, but most academic statisticians currently use R by default.

The opportunities for sharing ideas between statistics and ML are great and growing. The two papers presented here are important contributions in their own right, and also serve to highlight the potential benefits of narrowing the gap between the two communities. *It therefore gives me great pleasure to propose the vote of thanks.*

Conflict of interest: None declared.

References

Cranmer K., Brehmer J., & Louppe G. (2020). The frontier of simulation-based inference. *PNAS*, *117*(48), 30055–30062. https://doi.org/10.1073/pnas.1912789117

Shi Y., De Bortoli V., Deligiannidis G., & Doucet A. (2022). Conditional simulation using diffusion Schrödinger bridges. In J. Cussens and K. Zhang (Eds.), *Proceedings of the thirty-eighth conference on uncertainty in artificial intelligence: Vol. 180. Proceedings of machine learning research* (pp. 1792–1802). PMLR.

Song Y., Sohl-Dickstein J., Kingma D. P., Kumar A., Ermon S., & Poole B. (2021). 'Score-based generative modeling through stochastic differential equations', *International Conference on Learning Representations*. https://openreview.net/forum?id=PxTIG12RRHS

Seconder of the vote of thanks and contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Christopher Nemeth[1] (ORCID)

[1]Department of Mathematics and Statistics, Lancaster University, Bailrigg, Lancaster, LA1 4YW, UK

Address for correspondence: Christopher Nemeth, Department of Mathematics and Statistics, Lancaster University, Bailrigg, Lancaster, LA1 4YW, UK. Email: c.nemeth@lancaster.ac.uk

I congratulate the authors of these two papers for their insightful and significant contributions to addressing the statistical aspects of machine learning. In this contribution to these discussion papers, I will make a short comment which covers both papers and then provide some separate thoughts on each paper.

1 Statistical aspects of machine learning

These two papers are quite different in their focus; however, a common thread between them is the use of neural networks, and in particular deep neural networks, to augment part of the modelling process. In the case of Li et al. (2022), neural networks are used to convert the changepoint problem into a supervised learning problem, and in the case of Benton et al. (2022), neural networks are used to approximate the intractable score function.

A better understanding of the statistical properties of neural networks is an ongoing area of research (Anthony et al., 1999; Bartlett et al., 2019), but their application has become widespread within the artificial intelligence and machine learning communities. Within the statistics community, we can look back 30 years to the discussion paper of Ripley (1994) to see how neural networks can be used to solve classification problems. Interestingly, although the Ripley (1994) and Li et al. (2022) papers are very different in their focus, they both utilize neural networks to solve a classification problem and derive similar theoretical results regarding neural network complexity in terms of Vapnik–Chervonenkis (VC)-dimension bounds.

2 Paper 1: Automatic changepoint detection in time series via deep learning by Li et al.

The authors comment in the conclusion to their paper that for applied statisticians trying to model a changepoint problem: '…they need to understand what type of change is sought, be able to characterize it mathematically, find a satisfactory stochastic model for the data, formulate the appropriate statistic, and fine-tune its parameters'. However, if instead of trying to model the data-generating process, a neural network is used, then does this not lead to the same challenges? What type of neural network should be used? How deep should the network be? As an illustration of this point, let us consider the example from Section 5 of the paper, with $\rho = 0$ under Scenario 1. Figure 1 presented
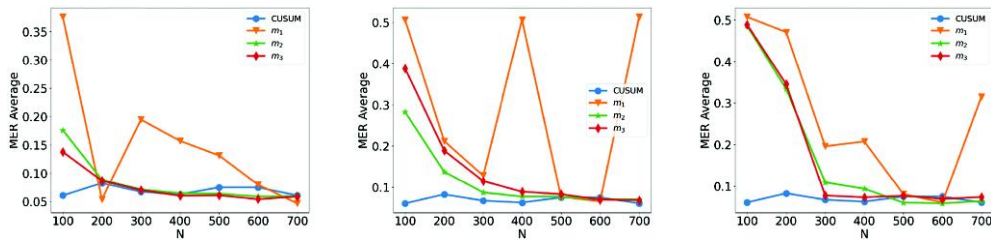
Figure 1. Scenario 1: $n = 100$, $N \in \{100, 200, \ldots, 700\}$, $\rho = 0$. ReLU (left), SeLU (middle), and Sigmoid (right) activation functions.

here is similar to Figure 2 in Li et al. (2022) where I have changed the activation function from ReLU (left) to SeLU (middle) and sigmoid (right). Under these different activation functions, we can see that for sufficiently large N, the neural network-based approaches are superior to the CUSUM statistics, but for small N the average misclassification error rate can be quite different depending on the choice of the activation function. What guidelines are available for practitioners to ensure that they use the best neural network architecture? Is it easier to choose an appropriate neural network than it is to choose a statistical model that directly models the time series data?

The authors focus their empirical presentation on the statistical improvements of their neural network-based approach when compared against the CUSUM statistic. However, there is no presentation of the difference in computational cost between the CUSUM and neural network approaches. Would it be more reasonable to report the misclassification error rate scaled by computational time? This would be interesting to consider because if it takes twice as long, or perhaps longer, to fit the neural network compared to the CUSUM test, then what percentage of improvement should we expect to see from the neural network as a result of the increased computational complexity?

3 Paper 2: From denoising diffusions to denoising Markov models by Benton et al.

The general denoising framework proposed in this paper is an important contribution that allows the class of diffusion generative models, which are rapidly growing in popularity, to be applied to non-Euclidean spaces. The examples in the paper are directed towards sampling from non-standard spaces; however, the first example, which considers an approximate Bayesian inference model, is on $\mathbb{R}^d$ and provides a nice illustration of how these diffusion models are applied on simpler problems. What is of particular interest in Section 6.1 is that the results presented are not significantly better than many existing approximate Bayesian computation algorithms. By some standards, this is quite a simple example as there are only four parameters to be learnt, and yet the neural network (a multilayer perceptron) used by the authors to approximate the score function has 1.9 million parameters (see Appendix J.1). It does seem somewhat paradoxical that in order to approximate a four-dimensional distribution it is necessary to estimate 1.9 million parameters in a neural network. Furthermore, the observed dataset is of size 250, which leads to interesting questions around how feasible it is to learn a large number of parameters from a neural network with small datasets. Is it possible to know what type of neural network should be used for a particular generative problem, e.g. images, text, etc.? Or how large the network needs to be in order to achieve high levels of statistical accuracy?

In conclusion, these two papers provide stimulating contributions to the field of statistical machine learning and open up many interesting avenues of future research in these areas, it is a pleasure to second the vote of thanks.

Conflict of interests: None declared.

References

Anthony M., Bartlett P. L., & Bartlett P. L. (1999). *Neural network learning: Theoretical foundations* (*Vol. 9*). Cambridge University Press Cambridge.

Bartlett P. L., Harvey N., Liaw C., & Mehrabian A. (2019). Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. *The Journal of Machine Learning Research*, 20(1), 2285–2301.

Benton J., Shi Y., De Bortoli V., Deligiannidis G., & Doucet A. (2022). 'From denoising diffusions to denoising Markov models', arXiv, arXiv:2211.03595, preprint: not peer reviewed.

Li J., Fearnhead P., Fryzlewicz P., & Wang T. (2022). 'Automatic change-point detection in time series via deep learning', arXiv, arXiv:2211.03860, preprint: not peer reviewed.

Ripley B. D. (1994). Neural networks and related methods for classification. *Journal of the Royal Statistical Society: Series B (Methodological)*, 56(3), 409–437. https://doi.org/10.1111/j.2517-6161.1994.tb01990.x

The vote of thanks was passed by acclamation.

M.N.M. van Lieshout and C. Lu's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Marie-Colette van Lieshout[1,2] 🔘 **and Changqing Lu**[2]

[1]CWI, Amsterdam, The Netherlands
[2]Department of Applied Mathematics, University of Twente, Enschede, The Netherlands

Address for correspondence: Marie-Colette van Lieshout, CWI, P.O. Box 94079, 1090GB Amsterdam, The Netherlands.
Email: Marie-Colette.van.Lieshout@cwi.nl

We congratulate Professors Li, Fearnhead, Fryzlewicz, and Wang on their fine work that represents classic test statistics for change-point detection as a neural network-based classifier and develops improved offline detection algorithms for historical, labelled data.

The paper focuses on real-valued observation vectors in a temporal regression model with training data being either labelled historical data or obtained by simulation from a model. In many applications, though, the data consist of both spatial and temporal components. For instance, the observations may take the form of a series of point patterns (e.g. mapped tree locations at different census times) or a single observation from a spatio-temporal point pattern (e.g. occurrence locations and times of fire incidents). For the latter, detection of changes in intensity in a model-based Bayesian test setting was investigated by Altieri et al. (2015). Do the authors believe that an adapted neural network approach could be competitive in this context? A complication would be that the change-point is due to complex changes in inter-point interaction for which neither a known model nor labelled historical data is available. Do the authors see a way forward here?

Machine learning ideas could benefit spatio-temporal statistical practice more widely. Specifically for point pattern analysis, Lu, Van Lieshout et al. (2023) employed random forest importance scores for variable selection, whilst Jalilian and Mateu (2023) trained neural networks on simulated data to distinguish spatial structural differences. A similar motivation as that of Li et al. is seen in point process intensity estimation. Usually, the intensity function is assumed to be log-linear in spatial and temporal covariates. Lu, Guan, et al. (2023) proposed a tree-based model, XGBoostPP, which forms the intensity function based on a covariate vector $z(s)$ as

$$\log\{\lambda(s)\} = \sum_{k=1}^{K} f_k\{z(s)\}.$$

Here, $f_k\{z(s)\}$ are tree predictors that output the response on the leaf where a covariate value $z(s)$ lies in. For model fitting, we customized a penalized weighted Poisson log-likelihood loss function

$$\sum_{k=1}^{K} \Omega(f_k) - \sum_{x \in X} w(x) \log\{\lambda(x)\} + \int_S w(s)\lambda(s)\mathrm{d}s$$

where X denotes the point process on S and $\Omega(f_k)$ is proportional to the L_1-norm of leaf responses. The tree structures and corresponding leaf responses are optimized iteratively; the weights w are calculated based on the estimated inhomogeneous K-function. The classic log-linear intensity function can be represented as a reparameterized XGBoostPP; neural networks may offer alternatives.

In the reverse direction, the stable with respect to iterations of tessellations tessellation (Nagel & Weiss, 2005) from spatial statistics can be used for partitioning responses (cf. Ge et al., 2019).

Conflict of interests: None declared.

References

Altieri L., Scott E. M., Cocchi D., & Illian J. B. (2015). A changepoint analysis of spatio-temporal point processes. *Spatial Statistics*, *14*(B), 197–207. https://doi.org/10.1016/j.spasta.2015.05.005

Ge S., Wang S., Teh Y. W., Wang L., & Elliott L. (2019). Random tessellation forests. In *NeurIPS 32*.

Jalilian A., & Mateu J. (2023). Assessing similarities between spatial point patterns with a Siamese neural network discriminant model. *Advances in Data Analysis and Classification*, *17*(1), 21–42. https://doi.org/10.1007/s11634-021-00485-0

Lu C., Guan Y., Van Lieshout M. N. M., & Xu G. (2023). *XGBoostPP: Tree-based estimation of point process intensity functions*, working paper.

Lu C., Van Lieshout M. N. M., De Graaf M., & Visscher P. (2023). Data-driven chimney fire risk prediction using machine learning and point process tools. *The Annals of Applied Statistics*, *17*(4), 3088–3111. https://doi.org/10.1214/23-AOAS1752

Nagel W., & Weiss V. (2005). Crack STIT tessellations: Characterization of stationary random tessellations stable with respect to iteration. *Advances in Applied Probability*, *37*(4), 859–883. https://doi.org/10.1239/aap/1134587744

Gilbert MacKenzie's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Gilbert MacKenzie (ID)

Formerly of: The Centre for Biostatistics, Department of Mathematics and Statistics, University of Limerick, Ireland and ENSAI, Rennes, France

Address for correspondence: Gilbert MacKenzie, 85 Maryville Park, Belfast BT9 6LQ, UK. Email: gilbert.mackenzie.cbs@gmail.com

Contribution

I would like first to congratulate both authors on presenting their very stimulating work in the spirit of the conference's theme of statisticians and machine learners working in alliance. There is no doubt in my mind that collaboration will lead to better, more parsimonious, models. My comments relate, mainly, to the second discussant's points on over-parametrization.

If a feed-forward neural net (Ripley, 1994) is denoted by $\mathcal{N}_{et}(x, \theta)$, where x is the input layer and ϕ is the defining parameter vector, colleagues, in the University of Limerick, cast the problem in a standard non-linear regression mould, thus:

$$Y_i = \mathcal{N}_{et}(x_i, \theta) + \epsilon_i,$$

where $\epsilon_i \sim N(0, \sigma^2)$ (McInerney & Burke, 2022a). This leads to standard Gaussian $\log_e$ likelihood, $\ell(\theta)$, for inference (here θ represents the collection of all net parameters to be estimated). Standard theory leads to $\hat{\theta} \sim N(\theta, \Sigma)$ where $\hat{\Sigma} = I_o(\hat{\theta})^{-1}$, where $I_o(\hat{\theta})$ is the observed, positive definite, information matrix, whence the uncertainty in θ can be assessed. Some care is required to avoid the singularities in $\hat{\Sigma}$ caused by over-fitting, e.g. when dealing with redundant nodes.

Given this probability model and its likelihood function, the model space is searched using the Bayesian information criterion. The choice of Bayesian information criterion, rather than the conventional out-of-sample performance criterion for model selection, leads to an increased probability of recovering the true (more parsimonious) model with comparable, or better, out-of-sample performance. A three-stage, backwards, model selection strategy is employed: it first reduces the number of hidden nodes (architectural pruning), then inputs (variable selection), and ends with a final adjustment (fine-tuning).

The availability of the variance–covariance matrix opens up opportunities to (a) frame the output of the neural net analysis in a more familiar, regression-like, format and (b) test for the existence of known or hypothetical structures in the data, for example, as in genetic analysis. There are also other modelling opportunities. Analyses carried out to date are promising; in some cases, comparable performance is obtained with half of the parameters required by the standard neural net.

This work is at an early stage and the neural nets studied so far are less complicated than those discussed today. However, as the project progresses (McInerney & Burke, 2022b), it aims to cover the neural net piste.

Meanwhile, we must not overlook the unresolved problem of 'explanation'.

Conflict of interests: None declared.

References

McInerney A., & Burke K. (2022a). 'A statistically-based approach to feedforward neural network model selection', arXiv, arXiv:2207.042 48v1, preprint: not peer reviewed.

McInerney A., & Burke K. (2022b). *Selectnn: A statistically-based approach to neural network model selection. R package version 0.0.0.9000*, https://github.com/andrew-mcinerney/selectnn

Ripley B. D. (1994). Neural networks and related methods for classification. *Journal of the Royal Statistical Society: Series B*, 56(3), 409–456. http://doi.org/10.1111/j.2517-6161.1994.tb01990.x

Bo Zhang's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Bo Zhang 🔵

International Institute of Finance, School of Management, University of Science and Technology of China, Hefei, China

Address for correspondence: Bo Zhang, 509, Building of Management Academy, East Campus of USTC. No. 96 Jinzhai Rd, Hefei 230026, Anhui Province, P.R. China. Email: wbchpmp@ustc.edu.cn

I congratulate Li, Fearnhead, Fryzlewicz, and Wang for their interesting contributions to change-point detection and deep learning. I wish to comment on the theoretical properties and potential extensions.

In Section 4, authors study the generalization error of neural network change-point classifiers based on the multivariate normal random vector $\mathbf{X} \sim N_n(\mu, \mathbf{I}_n)$. But the key of theoretical proofs is for any $1 \leq i \leq n - 1$,

$$P(|\mathbf{v}_i^\top \mathbf{X}| > t) \leq \exp(-t^2/2)$$

Thus, when $\mathbf{X}$ is sub-Gaussian, we can get similar results as Lemma 4.1 and Corollary 4.1. When $\mathbf{X}$ is heavy-tail, $P(|\mathbf{v}_i^\top \mathbf{X}| > t)$ is larger. It leads to a larger λ and a higher error in Corollary 4.1. On the other hand, the independence between $\mathbf{X} = (X_1, \ldots, X_n)$ is not necessary. $\mathbf{X}$ can be a time series with weak time dependence. Thus, the method proposed by authors works for real data.

Now we go to (4) in Theorem 4.3. Authors bound the generalization error by the sum of two parts. The first part is $\min_{h \in \mathcal{H}_{L,\mathbf{m}}} P(h(\mathbf{X}) \neq Y)$ and the second part depends on the complexity of the neural network class measured in its Vapnik–Chervonenkis (VC) dimension. Here $\mathcal{H}_{L,\mathbf{m}}$ is the class of functions with L hidden layers and width vector $\mathbf{m}$. When $\mathbf{X}$ has time dependence, the second part is more complicated to control. But we can follow the idea in Yu (1994) to deal with it.

The above framework of proofs is classical and transparent. However, it seems not perfect on studying and understanding neural networks. As we know, both N and L impact on the generalization error. But when we study the first term $\min_{h \in \mathcal{H}_{L,\mathbf{m}}} P(h(\mathbf{X}) \neq Y)$ separately, we can find that it does not depend on N. When L becomes larger, $\min_{h \in \mathcal{H}_{L,\mathbf{m}}} P(h(\mathbf{X}) \neq Y)$ should decrease. In contrast, the second term decreases when N increases and L decreases. Following the above thoughts, we have some conjectures:

When N is large enough, the second part for any neural networks is small enough but the first part for the neural network with larger L is smaller. Thus, the neural network with larger L outperforms others. In contrast, when N is small, the second part for the neural network with larger L is larger due to its higher VC dimension, and it may underperform.

Unfortunately, the above conjectures seem wrong. Figure 2 shows that the neural network with $L = 10$ outperforms others when N is small. In contrast, all neural networks perform similarly when N is large. Specifically, increasing L significantly reduces the average MER when $N \leq 200$ rather than $N \geq 500$. It is hard to understand this phenomenon by the framework in Theorem 4.3. Thus, seeking some new frameworks on bounding the generalization error is meaningful. It may help us understand the impact of L better.

Conflict of interest: None declared.

Reference

Yu B. (1994). Rates of convergence for empirical processes of stationary mixing sequences. *The Annals of Probability*, 22(1), 94–116. https://doi.org/10.1214/aop/1176988849

Andi Wang's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Andi Q. Wang [ORCID]

Department of Statistics, University of Warwick, Coventry, UK

Address for correspondence: Andi Q. Wang, Department of Statistics, University of Warwick, Coventry, CV4 7AL, UK. Email: andi.wang@warwick.ac.uk

I would like to congratulate the authors for this excellent article, which includes both careful theoretical developments at the foundations of such denoising models and exciting methodological innovations and experiments. Of course, there are still many unanswered questions in relation to these denoising Markov models. One of the most tantalizing such questions is touched on by the authors in their concluding discussion, namely the nature of the implicit biases which are present in this methodology.

It appears to me that there are (at least) two potential sources of implicit bias in such denoising models. First, the use of the evidence lower bound (ELBO) training objective. Although the use of the ELBO is introduced as a concession to the fact that the true likelihood $p_T(\mathbf{x})$ is intractable, it has been shown in some cases that using an ELBO actually has beneficial implicit biases. For example, in the context of variational auto-encoders (VAEs), it has been show in certain settings that the use of the ELBO actually enables the VAE to disentangle independent signals (Reizinger et al., 2022). Although for score matching we can see the minimizer is $s_\theta(\mathbf{x}, t) = \nabla_{\mathbf{x}} \log q_t(\mathbf{x})$, I wonder if this particular formulation of the objective leads to some beneficial implicit biases. Second, it is well known in the context of supervised learning that (stochastic) gradient descent often leads to solutions which generalize well (Chizat & Bach, 2020; Soudry et al., 2017)—it is thus natural to ask if a similar phenomenon is occurring in the denoising Markov models setting. Thus one wonders if, in conjunction with the ELBO, the use of gradient descent in this context is resulting in score estimates which 'generalize' well, providing novel samples rather than recovering the empirical distribution of the training data. And of course, the particular choice of neural network architecture and network hyperparameters will no doubt further bias the score estimate towards certain approximations.

Overall, it would be fascinating to get a handle on which components or combination of components of the methodology are the main contributors towards the very impressive empirical performance. I would like to congratulate the authors again for this fantastic contribution to the literature and heartily agree that further attention is required!

Conflict of interest: None declared.

References

Chizat L., & Bach F. (2020). Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. *Proceedings of Thirty Third Conference on Learning Theory*, 125, 1305–1338. https://doi.org/10.48550/arXiv.2002.04486

Reizinger P., Gresele L., Brady J., von Kügelgen J., Zietlow D., Schölkopf B., Martius G., Brendel W., & Besserve M. (2022). Embrace the gap: VAEs perform independent mechanism analysis. *Advances in Neural Information Processing Systems*, 35, 12040–12057. https://doi.org/10.48550/arXiv.2206.02416

Soudry D., Hoffer E., Nacson M. S., Gunasekar S., & Srebro N. (2017). The implicit bias of gradient descent on separable data. *Journal of Machine Learning Research*, 19, 1–57. https://doi.org/10.48550/arXiv.1710.10345

Kanti Mardia's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Kanti V. Mardia[1,2] 🆔

[1]Department of Statistics, University of Leeds, Leeds, UK
[2]Department of Statistics, University of Oxford, Oxford, UK

Address for correspondence: Kanti V. Mardia, Department of Statistics, University of Leeds, Leeds, LS2 9JT, UK. Email: k.v.mardia@leeds.ac.uk

I enjoyed reading this paper which has so many outstanding features; this is where Modern Statistics is moving—high-level computing with sound mathematical statistical. My comments are as follows.

The authors use the wrapped normal on $SO(3)$ using the exponential map and I have some reservations related to this choice since the most common distribution used on $SO(3)$ is the Fisher matrix distribution which has many desirable properties (see, e.g. Mardia & Jupp, 2000) whereas there are inherent singularities in this type of wrapping which I will illustrate. Since $SO(3)$ can be identified with S^3 (after identifying antipodal points), any calculation on $SO(3)$ can be reformulated as one on S^3 and will consider the unit sphere S^{q-1}. If $g(x)$ is a pdf in the tangent space, x in $\mathbb{R}^{q-1}$, then the 'wrapped' pdf (with the exponential map on the sphere with base point at the north pole $(0, \ldots, 0, 1)$) of $y = (\sin\theta v, \cos\theta)$, where $0 \leq \theta \leq \pi$ and v is a unit vector in $\mathbb{R}^{q-1}$, can be shown to have the pdf

$$f(y) = (1/\sin^{q-2}\theta) \sum_{k=0}^{\infty} \{r_{1,k}^{q-2} g(r_{1,k}v) + r_{2,k}^{q-2} g(-r_{2,k}v)\}$$

with respect to the uniform measure on the sphere, where $r_{1,k} = \theta + 2\pi k$ and $r_{2,k} = 2\pi(k+1) - \theta$. Except for the term involving $r_{1,0}$ at $\theta = 0$, all the remaining terms have a singularity at $\theta = 0$ and at $\theta = \pi$. In particular, this wrapped distribution on the sphere cannot be unimodal. This is not a great problem if $g(\cdot)$ is highly concentrated near the origin on the tangent plane, but it can be an issue for more diffuse distributions (and particularly when used in mixtures, Section J5). I note that the construction is general: given a base point m on the Riemannian manifold and a vector x in the tangent space, the exponential map $\exp_m(x)$ yields a point on the manifold and we can go from $g(\cdot)$ to $f(\cdot)$.

The use of the score matching methodology (SME) methodology is remarkable in our work (Mardia et al., 2016), which you quote on manifolds has been more in the classical statistical setting. On the other hand, the introduction of score matching approximation in Mardia (2018) explores totally a new direction; how to approximate 'analytically' a known distribution f^* by another simpler distribution f from an exponential family in contrast to your data-driven method. A general solution is given in this paper: the score matching approximation (SMA) is motivated by the challenging problem of approximating the standard multivariate wrapped normal distribution by the multivariate von Mises distribution.

My last point is on your examples. Your emphasis on assessing the performance of your methods is via the ground truths but perhaps in some cases such as for estimation, this may not be sufficient. I have also some queries on your example in Section J5: how did you make the choice of the number of components of the mixture $M = 16$?; it is also not clear how much noise was added for the test example via σ^2 and its effect; and are the standard errors of your estimates relevant?

Conflict of interests: None declared.

References

Mardia K. V. (2018). A new breakthrough in the estimation methodology for standard directional distributions. In *Proceedings of the 20th international conference on information fusion, IEEE, fusion*.

Mardia K. V., & Jupp P. E. (2000). *Directional statistics*. Wiley.

Mardia K. V., Kent J. T., & Laha A. K. (2016). *Score matching estimators for directional distributions*. arXiv preprint arXiv:1604.08470.

Shakeel Gavioli-Akilagun's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Shakeel Gavioli-Akilagun Sr.

Department of Statistics, London School of Economics, London, UK

Address for correspondence: Shakeel Gavioli-Akilagun, Department of Statistics, London School of Economics, London, UK. Email: s.a.gavioli-akilagun@lse.ac.uk

We congratulate the authors for the interesting and thought provoking paper. Lemma 3.2 shows that the generalized likelihood ratio (GLR) test for stability of a linear regression can be viewed as a simple feed forward neural network. However, close inspection of the lemma reveals that the setup rules out several common change point problems. Consider the piecewise polynomial regression:

$$y_i = \begin{cases} \sum_{j=0}^{p} \alpha_j (i/n - \tau/n)^j + \xi_i & \text{if } t \le \tau, \\ \\ \sum_{j=0}^{p} \beta_j (i/n - \tau/n)^j + \xi_i & \text{if } t > \tau, \end{cases} \quad i = 1, \ldots, n \quad (1)$$

For ξ's distributed i.i.d. $\mathcal{N}(0, 1)$, the likelihood ratio statistic for a change at location i is

$$\mathcal{R}_i(\mathbf{Y}) = \|P_{1:i}\mathbf{Y}\|^2 + \|P_{(i+1):n}\mathbf{Y}\|^2 - \|P_{1:n}\mathbf{Y}\|^2 \quad (2)$$

where $P_{s:e}\mathbf{Y}$ denotes the projection of elements indexed $\{s, \ldots, e\}$ in the vector $\mathbf{Y} = (y_1, \ldots, y_n)'$ onto the space of (discretised) polynomials of degree p. Since (2) is a linear combination of

quadratic forms, $h_\lambda^{\text{GLR}}(y) = \mathbf{1}_{\{\max_i \mathcal{R}_i(y) > \lambda\}}$ clearly cannot be represented as a neural network. The Wald test for the same problem (e.g. Kim et al., 2022) likewise cannot be represented in this way.

In Gavioli-Akilagun and Fryzlewicz (2023), we introduce tests based on differences of local sums of the data as simple and computationally efficient alternatives to GLR and Wald tests. Interestingly, our difference-based tests can be represented as a neural network. Consider the statistic

$$\mathcal{D}_i(\mathbf{Y}) = \left\{ \sum_{j=0}^{p+1} \binom{p+1}{j}^2 \right\}^{-1/2} \sum_{k=0}^{p+1} (-1)^{p+1-k} \binom{p+1}{k} \left[\frac{y_{i+(k-1)l_i+1} + \cdots + y_{i+kl_i}}{\sqrt{l_i}} \right]$$

where $l_i = \max\{l \in \mathbb{Z} \mid i - l \geq 0 \text{ and } i + (p+1)l \leq n\}$. Since $\mathcal{D}(\cdot)$ is a linear operator, $h_\lambda^{\text{DIF}}(\mathbf{x}) = \mathbf{1}_{\{\max_i |\mathcal{D}_i(\mathbf{x})| > \lambda\}}$ can be represented as a neural network. Using the techniques in Gavioli-Akilagun and Fryzlewicz (2023), one can show that the localization rate of Algorithm 1 for the change point in (1) is of the order

$$\mathcal{O}\left(\frac{B^2 n^{\frac{2p^*}{2p^*+1}}}{\Delta_{p^*}^2} \right)$$

where $\Delta_j = (\alpha_j - \beta_j)$, $p^* \in \arg\max_{j=0,\dots,p}\{|\Delta_j|(\delta/n)^j\}$, and $\delta = \tau \wedge (n - \tau)$. This is unimprovable up to the B^2 term. When analysing the behaviour of neural networks on change point problems, it may be useful to think in terms of difference-based tests.

Conflict of interest: None declared.

References

Gavioli-Akilagun S., & Fryzlewicz P. (2023). 'Fast and optimal inference for change points in piecewise polynomials via differencing', arXiv, arXiv:2307.03639, preprint: not peer reviewed.

Kim J., Oh H.-S., & Cho H. (2022). 'Moving sum procedure for change point detection under piecewise linearity', arXiv, arXiv:2208.04900, preprint: not peer reviewed.

The following contributions were received in writing after the meeting.

Ivor Cribben and Anastasiou Andreas's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Andreas Anastasiou[1] and Ivor Cribben[2]

[1]Department of Mathematics and Statistics, University of Cyprus, Nicosia, Cyprus
[2]Department of Accounting and Business Analytics, Alberta School of Business, University of Alberta, Edmonton, Canada

Address for correspondence: Ivor Cribben, Department of Accounting and Business Analytics, Alberta School of Business, University of Alberta, 11211 Saskatchewan Dr NW, Edmonton, Alberta T6G 2R6, Canada. Email: cribben@ualberta.ca

1 Introduction

We extend our congratulations to the authors for their innovative contribution. Here, we address three key points: change-point labelling, handling imbalanced datasets, and computational complexity.

Change-point labelling in datasets has many flaws and challenges. The primary flaw is the subjectivity inherent in the task as it often relies on human judgement. This subjectivity introduces biases and inconsistencies. One challenge is the lack of a universally accepted standard for change-point labelling unlike other machine learning classification problems. Hence, it is difficult to compare results across studies, hindering reproducibility and reliability. A further issue is that change-point labelling can be a time-consuming and labour-intensive task, especially for large and complex time series datasets. This process often requires domain expertise and can be impractical for real-time or high-frequency data analysis.

Imbalanced class distributions in datasets are another issue. Change-points are often rarer than normal instances, but imbalanced datasets can lead to skewed evaluation results, with methods prioritizing the majority class and failing to effectively detect true change-points or managing an excessive rate of false positives. We wonder whether the authors explored examples of imbalanced data, specifically those involving significant changes in approximately half of the dataset ($N/2$). It is important to underscore that addressing the labelling and imbalance challenges is pivotal for change-point methods that rely on training neural networks.

The sample size used for training neural networks plays a crucial role in determining the model's performance and generalizability. An excessively large sample size might lead to increased computational costs and training time without significant gains in performance after a certain point. In the first step of the proposed algorithm, there is the necessity of training a neural network using a considerable sample size. A discussion on this topic could convince the reader that the proposed method can be extended to an online framework (as discussed in Section 7). Apart from accuracy, and especially in high-frequency data, online change-point detection methods have to be very computationally efficient in order for users to act promptly. The computational complexity in the simple univariate setting is also crucial to understand extensions to practically meaningful adaptations of the algorithm to multivariate, possibly high-dimensional frameworks. Furthermore, an expansion of the method to the multiple change-point framework is discussed through an idea similar to that of moving sum (MOSUM; Eichinger & Kirch, 2018). It would be beneficial to the reader for the authors to justify this choice; is it due to MOSUM's low computational complexity?

Conflicts of interest: None declared.

Reference

Eichinger, B., & Kirch, C. (2018). A MOSUM procedure for the estimation of multiple random change points. *Bernoulli*, 24(1), 526–564. https://doi.org/10.3150/16-BEJ887

Dean Bodenham and Niall Adams's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Dean Bodenham 🆔 and Niall Adams

Department of Mathematics, Imperial College London, London, UK

Address for correspondence: Dean Bodenham, Department of Mathematics, Imperial College London, South Kensington Campus, London SW7 2AZ, UK. Email: dean.bodenham@imperial.ac.uk

We congratulate the authors for their thought-provoking paper and innovative approach for detecting changepoints in a supervised manner.

The results in Table 1 and Figure 2 show that the proposed approach has good performance in terms of the misclassification error rate metric, or in other words in determining whether or not a sequence contains a changepoint.

However, since a primary concern in changepoint detection is the accuracy of the localization of the changepoint, it would be interesting to see the performance of the proposed approach for offline changepoint detection metrics such as the covering metric (Arbelaez et al., 2010), where the locations of the detected changepoints are compared with the locations of the true changepoints (van den Burg & Williams, 2020). It would also be instructive to see this performance in comparison to established offline methods such as the pruned exact linear time method (Killick et al., 2012) or the wild binary segmentation method (Fryzlewicz, 2014), rather than the online cumulative sum method.

Another concern is how the proposed method would handle very long sequences, for example, of length more than one million. Besides computational challenges, one might expect most methods to flag a changepoint in such sequences, while the exact location of the changepoint(s) may be more difficult to determine.

Finally, we suggest that one potential area of application for the proposed method may be in cellular biology where cells are tracked and their protein expression levels are recorded. These time series are often short in length, potentially $N \leq 20$, which can be a challenging scenario for traditional, unsupervised changepoint detection methods, although for this application it may be possible to obtain labelled examples from experts for the necessary training.

We look forward to future developments of the proposed approach.

Conflict of interest: None declared.

References

Arbelaez P., Maire M., Fowlkes C. C., & Malik J. (2010). Contour detection and hierarchical image segmentation. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, *33*(5), 898–916. https://doi.org/10.1109/TPAMI.2010.161

Fryzlewicz P. (2014). Wild binary segmentation for multiple change-point detection. *The Annals of Statistics*, *42*(6), 2243–2281. https://doi.org/10.1214/14-AOS1245

Killick R., Fearnhead P., & Eckley I. A. (2012). Optimal detection of changepoints with a linear computational cost. *Journal of the American Statistical Association*, *107*(500), 1590–1598. https://doi.org/10.1080/01621459.2012.737745

van den Burg G. J. J., & Williams C. K. I. (2020). 'An evaluation of change point detection algorithms', arXiv, arXiv:2003.06222, preprint.

Yudong Chen and Yining Chen's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Yudong Chen ⓘ **and Yining Chen**

Department of Statistics, London School of Economics and Political Science, London, UK

Address for correspondence: Yudong Chen, Department of Statistics, London School of Economics and Political Science, Houghton Street, London, WC2A 2AE, UK. Email: y.chen276@lse.ac.uk

We congratulate the authors for providing this stimulating work which uses deep neural networks for detecting change-points. We shall comment on three aspects of the paper: (i) the standardization procedure, (ii) distributional assumptions of the training and testing samples, and (iii) potentially more efficient use of the training samples.

First, we emphasize that classification procedures based on neural networks will not automatically be invariant to shifting or scaling. To illustrate this point, we consider scenario S1 with $\rho_t = 0.5$ and with no standardization performed on the training or testing sets. Figure 1a shows the results where an independent $U \sim U[-2,2]$ mean-shift is added to each series from only the testing set. Without standardization, the proposed method is no longer desirable even when we increase the size of the training set, N, to 1,600. Scaling to [0,1], as proposed in the paper, will alleviate this issue. Standardization using trimmed mean and trimmed standard deviation estimates would be a more robust option. From our experiments, other approaches such as adding a random baseline to all training samples would also work.

Nevertheless, if the distributions of the training and test samples, denoted by D_{train} and D, are indeed the same, then performing standardization might lead to worse performance, as is illustrated in Figure 1b and 1c. We suspect that it is because of certain distributions change-point can be characterized by statistics that are not invariant to shifting or scaling. These statistics are likely simpler to learn than the Cumulative Sum (CUSUM). Although, as demonstrated before, without standardization, the resulting classifier might not be transferable to even slightly different settings.

Besides, the presented theory requires $D_{train} = D$. We anticipate similar results to hold if D_{train} is a finite mixture with one component being D. More broadly speaking, consistency should hold if the support of D_{train} contains that of D. In addition, one could use different labels for different types of change-points (e.g. change in mean/variance/etc.) in the training set, and apply the existing approach to learn a multi-class classifier.

Finally, we believe that the training samples can be used in a more efficient manner with a minor twist. For $X_1, X_2, \ldots, X_n$ with label Y, its *reversed* sequence $X_n, X_{n-1}, \ldots, X_1$ should also have the same label. Consequently, we could double the size of N by adding all the reversed sequences into the training set. Sizeable improvement with this extra step can be seen, especially when N is small, by comparing Figure 1d with Figure 1e.

Conflicts of interest: None declared.

(a)

$\mu_L = 0$ for the training samples, $\mu_L \sim U[-2,2]$ for the testing samples; data not standardized

(b)

$\mu_L = 0$ for both training and testing; data not standardized

(c)

$\mu_L = 0$ for both training and testing; data standardized

(d)

$\mu_L = 0$ for both training and testing; data standardized; no reversed samples added

(e)

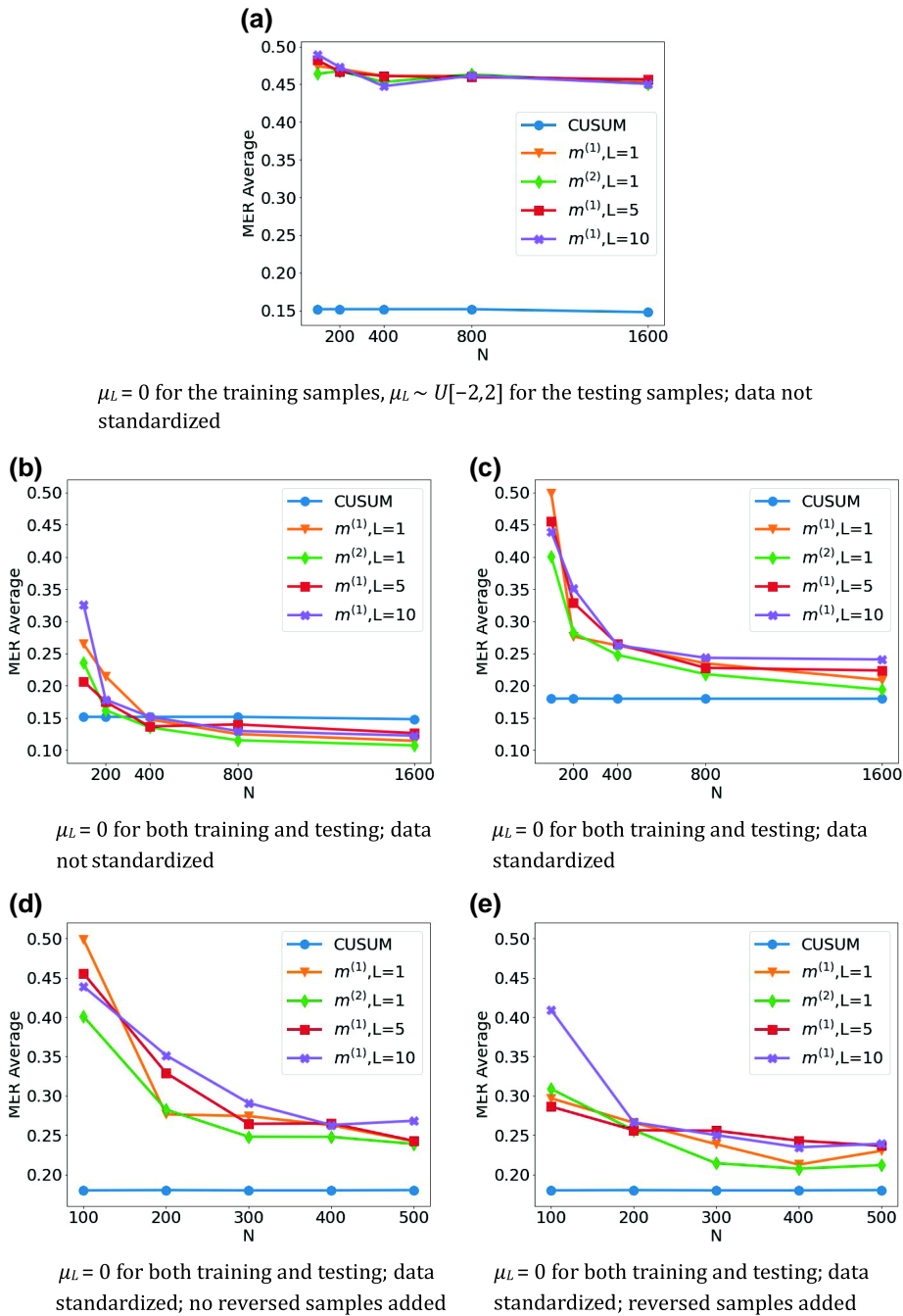$\mu_L = 0$ for both training and testing; data standardized; reversed samples added

Figure 1. Plot of test set misclassification error rate, computed on a test set of size 150,000, against training sample size n for detecting the existence of a change-point on data series of length $n = 100$ under scenario S1 described in Section 5 of the paper, with the exception that $\rho_t = 0.5$. We compare the performance of the CUSUM test and neural networks from four function classes as specified in Section 5. Here we use batch size of 32 for training with no regularization. For standardization, we use the approach suggested in the paper. For the testing set used to produce (a), an independent mean-shift of $U[-2,2]$ is added to each series. (a) $\mu_L = 0$ for the training samples, $\mu_L \sim U[-2,2]$ for the testing samples; data not standardized. (b) $\mu_L = 0$ for both training and testing; data not standardized. (c) $\mu_L = 0$ for both training and testing; data standardized. (d) $\mu_L = 0$ for both training and testing; data standardized; no reversed samples added. (e) $\mu_L = 0$ for both training and testing; data standardized; reversed samples added.

Daniela Cialfi's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Daniela Cialfi[1,2] ◉

[1]Institute for Complex Systems—National Research Council of Italy, UoC La Sapienza University, Rome, Italy
[2]Enrico Fermi Research Center, Rome, Italy

Address for correspondence: Daniela Cialfi, Centro Ricerche Enrico Fermi, Piazza del Viminale, 1, I-00184 Rome, Italy; Istituto dei Sistemi Complessi (ISC) - CNR, UoS Sapienza, P.le A. Moro, 2, I-00185 Rome, Italy. Email: danielacialfi@gmail.com

From Denoising Diffusions to Denoising Markov Models—An Excited New Perspective?

The paper entitled *From Denoising Diffusions to Denoising Markov Models* explores how the denoising diffusion models can be used in twofold aspects: (i) work by diffusing the data distribution into a Gaussian one, learning, at the same time, to reverse this noising process to obtain synthetic data points and (ii) perform approximate posterior simulation when we are in the presence of a sample operation from the prior and likelihood. Furthermore, the authors propose a unifying framework generalizing this approach to a wide class of spaces and leading to an original extension of score matching. We illustrate the resulting models on various applications.

From the above summary and after having carefully read the paper, it is possible to affirm that it explains the concept inside very convincingly, but there is a way to improve it. In particular, it is possible to suggest testing this new and fascinating denoising Markov models into a real scenario, such as theoretical neurobiology in which some of the concepts, like elaborate model use, are applied and discussed under their formal mathematical aspects.

Conflicts of interest: None declared.

Pierre-Aurelien Gilliot, Christophe Andrieu, Anthony Lee, Song Liu, and Michael Whitehouse's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Pierre-Aurelien Gilliot ◉**, Christophe Andrieu, Anthony Lee, Song Liu and Michael Whitehouse**

School of Mathematics, University of Bristol, Bristol, UK

Address for correspondence: Pierre-Aurelien Gilliot, School of Mathematics, University of Bristol, Bristol BS8 1QU, UK. Email: ys18223@bristol.ac.uk
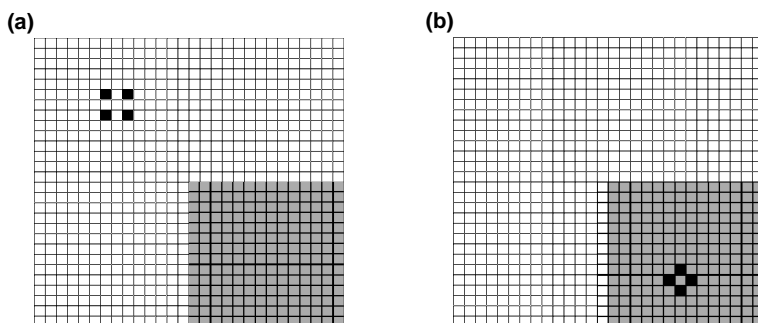
Figure 1. Configurations in the training dataset: a) squares and b) diamonds. The grey shading represents the bottom right quadrant. Images are binarised using a threshold set at 0.5 for easier visualisation.
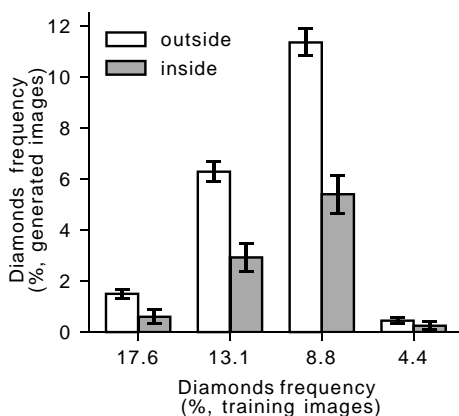

Figure 2. Frequency of diamonds. The grey bar represents the frequency of diamonds generated inside the bottom right quadrant while white bar indicates the frequency of diamonds generated outside of it.

We would like to thank the authors for their interesting contribution and the clarification brought on a particular aspect of diffusion models concerned with the criterion used in the fitting phase of the procedure. Another aspect, not discussed extensively, is the choice of the neural network used to model the gradients involved in the generative phase. In particular, the implicit prior information this induces on the class of distributions considered may seem mysterious and perhaps not always controllable? Here, we report results from a toy numerical experiment, which seem to raise some questions.

We consider 28×28 images, in which all but four pixels have values distributed according to a Beta(10,10), re-scaled between 0 and 0.49. The four remaining pixels have values distributed according to a Beta(10,10), re-scaled between 0.51 and 1, and are organised in two distinct configurations: square or diamond (Figure 1). Squares are uniformly distributed across the whole image, while diamonds are uniformly distributed only in the bottom right quadrant. Four different datasets were created, each comprising 82,000 of such images, with varying proportions of squares and diamonds (Figure 2), and were each used to train four different U-nets approximating the score function following standard practice (Ho et al., 2020; HuggingFace, 2023).

A generative model which is coherent with this dataset should be expected to generate images with squares or diamonds, respecting pixel boundaries for the diamonds, and matching the frequencies of each configuration.

Remarkably, most sampled images (94%) correctly consist of either squares or diamonds. Diamonds were not confined to the bottom right quadrant anymore (Figure 2), which could stem from the shift-equivariant property of the U-Net's convolutional layers (Cohen & Welling, 2016). However, this may not be a desirable feature.

Diffusion modelling aims at alleviating the challenges faced by score matching when dealing with disjointed, multimodal distributions such as the mixture square/diamond considered here (Song & Ermon, 2019). However, we encountered difficulties in accurately sampling according to the training set mixture weights: the sampled diamond frequencies did not reflect the training set diamond frequencies across the various datasets (Figure 2). Interestingly, these sampled frequencies differed when using another seed to initialise the U-Net weights (but were still unexpected).

While the link between simple properties of the neural network chosen and the class of distributions implicitly defined can sometimes be understood (Yim et al., 2023), a general characterisation is missing from the literature. This phenomenon is further compounded by additional unknown neural network properties. We are curious about the current understanding of this matter.

Conflict of interest: None declared.

References

Cohen T.S., & Welling M. (2016). 'Group equivariant convolutional networks', arXiv, arXiv:1602.07576, preprint: not peer reviewed.

Ho J., Jain A., & Abbeel P. (2020). 'Denoising diffusion probabilistic models', arXiv, arXiv:1602.07576, preprint: not peer reviewed.

HuggingFace (2023). blog/annotated-diffusion.md at main. huggingface/blog—github.com. https://github.com/huggingface/blog/blob/main/annotated-diffusion.md. accessed October 5, 2023.

Song Y., & Ermon S. (2019). 'Generative modeling by estimating gradients of the data distribution', arXiv, arXiv:1907.05600, preprint: not peer reviewed.

Yim J., Trippe B.L., De Bortoli V., Mathieu E., Doucet A., Barzilay R., & Jaakkola T. (2023). Se(3) diffusion model with application to protein backbone generation.

Yongmiao Hong, Oliver Linton, Jiajing Sun, and Meiting Zhu's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Yongmiao Hong[1,2], Oliver Linton[3], Jiajing Sun[2,4] 🆔 and Meiting Zhu[5]

[1]Center for Forecasting Science, Chinese Academy of Sciences, Beijing, China
[2]School of Economics and Management, and MOE Social Science Laboratory of Digital Economic Forecasts and Policy Simulation, University of Chinese Academy of Sciences, Beijing, China
[3]Faculty of Economics, University of Cambridge, Cambridge, United Kingdom
[4]School of Mathematics, University of Birmingham, Birmingham, United Kingdom
[5]School of Economics, Xiamen University, Fujian Province, China

Address for correspondence: Jiajing Sun, School of Economics and Management, and MOE Social Science Laboratory of Digital Economic Forecasts and Policy Simulation, University of Chinese Academy of Sciences, No. 3, Zhongguancun South First Street, Haidian District, Beijing, 100190, China. Email: jiajing.sun@gmail.com

This paper proposes a neural network-based approach for automating offline change-point detection. The authors show that cumulative sum (CUSUM) and generalized CUSUM are a special case of their neural network class. They emphasize misclassification error rates and their theoretical contribution is to establish some elegant results for these under i.i.d. unit variance Gaussian data with a possible change in mean. Their theoretical results outline the conditions on the empirical risk minimization neural network that allow it to achieve comparable performance to the classic CUSUM test for this very specific setting. The framework relies on N training data samples that are independent and identical copies where $N \gg n^2 \log n$, which seems like a lot of training is needed! The CUSUM test only needs the sample of size n. In many financial applications, we have a single time series $\{X_1, \ldots, X_n\}$ and we are interested in when and how change points occur throughout the whole observation period. It is not clear how or why we should divide the data into training and testing samples and some guidance on this would be appreciated. In those applications, considerable care needs to be taken in how to account for time series dependence and where this is treated non-parametrically issues arise with bandwidth selection. The self-normalization method proposed by Shao (2010) does not require the choice of tuning parameter and so is also automatic, and is quite widely used for identifying change points within a given dataset (Shao & Zhang, 2010). However, this method is known to have poor power properties. Recently, Hong et al. (2024) proposed the adjusted-range-based self-normalization, instead of the usual long-run variance normalization, and this appears to work better under some long memory alternatives.

The current paper avoids the normalization issue altogether by making use of training data, and perhaps where that exists their method can have advantages, but it would be interesting to know how competitive their method is on financial time series and on realistic sampling schemes with longer range dependence.

Conflict of interest: The authors declare that there are no conflicts of interest regarding the publication of this paper.

Funding

References

Hong, Y., Linton, O., McCabe, B., Sun, J., & Wang, S. (2024). Kolmogorov–Smirnov type testing for structural breaks: A new adjusted-range based self-normalization approach. *Journal of Econometrics*, *238*(3), 105603. https://doi.org/10.1016/j.jeconom.2023.105603

Shao X. (2010). A self-normalized approach to confidence interval construction in time series. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *72*(3), 343–366. https://doi.org/10.1111/j.1467-9868.2009.00737.x

Shao X., & Zhang X. (2010). Testing for change points in time series. *Journal of the American Statistical Association*, *105*(491), 1228–1240. https://doi.org/10.1198/jasa.2010.tm10103

James Jackson's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

James Jackson [ID]

The Alan Turing Institute, London, UK

Address for correspondence: James Jackson, The Alan Turing Institute, The British Library, 96 Euston Road, London, UK. Email: jjackson@turing.ac.uk

Relating to the excellent Benton et al. paper, I just wanted to highlight a potential use of denoising Markov models (DMMs): the generation of synthetic data sets for statistical disclosure control (Little, 1993; Rubin, 1993). Although this link to synthetic data is somewhat obvious—after all, the objective of generative modelling is essentially equivalent to that of synthetic data generation—such crossovers between different areas of statistics (never mind between statistics and machine learning) are often overlooked.

The synthetic data literature (e.g. Drechsler, 2011) is naturally focused on the $\mathbb{R}^d$ space (or, say, $\{0, 1\}^d$ if the data are binary). It would be interesting to see: (i) how DMMs perform relative to traditional synthetic data methods (e.g. the use of generalized linear models), not only in terms of risk and utility, but also in terms of computational time; and (ii) how DMMs perform when synthesizing data sets on more complex spaces, where traditional synthetic data methods fail.

Statistics and machine learning are becoming ever more intertwined. The generation of synthetic data is a good example of this and DMMs can be added into the ever-growing array of synthetic data generation methods.

Conflict of interest: None declared.

References

Drechsler, J. (2011). *Synthetic datasets for statistical disclosure control: Theory and implementation*. Springer.
Little R. J. (1993). Statistical analysis of masked data. *Journal of Official Statistics*, 9(2), 407–426.
Rubin D. B. (1993). Statistical disclosure limitation. *Journal of Official Statistics*, 9(2), 461–468.

Ayla Jungbluth and Johannes Lederer's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Ayla Jungbluth [ID] and **Johannes Lederer**

Department of Mathematics, Ruhr University Bochum, Bochum, Germany

Address for correspondence: Ayla Jungbluth, Department of Mathematics, Universitätsstraße 150, Ruhr University Bochum, 44801 Bochum, Germany. Email: ayla.jungbluth@rub.de
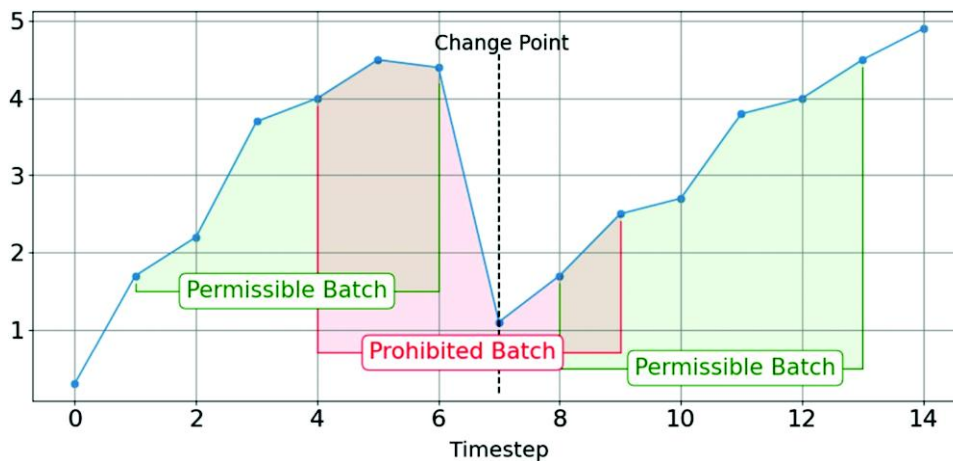
Figure 1. Visualization of the change-point method *BatchCP, taken from the mentioned reference*. It shows an example of a batch selection around a change-point. The points represent time-series data. The first area shows a batch created by choosing $t = 1$ as the start index, and with a selected batch size of $s = 6$, the batch contains the points $t = 1$ to $t = 6$. The change-point is located at $t = 7$ and is therefore outside the detected batch. The batch is therefore permissible. The area in the middle shows a prohibited batch. The start index of this batch is $t = 4$ and the end index at $t = 9$, so the change-point is located inside the batch and is not allowed in training. A new batch must be found. The area to the right indicates another permissible batch, since the change-point lies outside of it.

Introduction

We have read the paper 'Automatic Change-Point Detection in Time Series Via Deep Learning' by Jie Li, Paul Fearnhead, Piotr Fryzlewicz, and Tengyao Wang with great interest. We congratulate the authors for an important and forward-looking contribution. We agree that time-series data often have change points that can degrade the prediction quality considerably. The paper's use of deep learning to detect change points is, therefore, definitely interesting.

Besides, our own work shows that change-point detection is also vital for time-series pipelines that are based on deep learning themselves (Jungbluth & Lederer, 2023). We have also shown that change points can be accounted in deep learning in a surprisingly simple and effective way. Indeed, most deep-learning-based forecasting methods learn by taking batches from the time-series data. In essence, our BatchCP method selects batches from the time series such that change points are avoided while we are still able to use the entire time series as data. For this, BatchCP needs to know where the change points are before the training begins. Based on this information, the batches are then deemed suitable or not as illustrated in Figure 1.

BatchCP is agnostic to the way change points are detected and, therefore, should work very well with the method proposed in the discussed paper. Indeed, the combination of BatchCP and the proposed method in their paper would equip modern architectures such as DeepAR and transformers with a quite automated approach to deal with change points. Our numerical studies still need to be completed, but the initial results look promising already.

Conflicts of interest: None declared.

Reference

Jungbluth, A., & Lederer, J. (2023). 'The DeepCAR Method: Forecasting Time-Series Data That Have Change Points', arXiv:2302.11241, preprint: not peer reviewed.

John Kent's contribution to the Discussion of the 'Discussion Meeting on Probabilistic and statistical aspects of machine learning'

John T. Kent (ID)

School of Mathematics, University of Leeds, Leeds, UK

Address for correspondence: John T. Kent, School of Mathematics, University of Leeds, Woodhouse Lane, Leeds LS2 9JT, UK. Email: j.t.kent@leeds.ac.uk

I would like to thank Benton and his co-authors for a very stimulating paper. I will focus my comments on some ideas related to diffusions and directional distributions.

In the context of Euclidean space, the forwards diffusion in the discussion below equation (2) is taken to be an Ornstein–Uhlenbeck process with drift $b(\boldsymbol{x}) = -\lambda \boldsymbol{x}$, and with infinitesimal variance equal to σ^2. The equilibrium distribution is an isotropic multivariate normal distribution, $N_p(\boldsymbol{0}, \sigma^2/(2\lambda)I)$. The authors take $\lambda = 1/2$ and $\sigma^2 = 1$, but I wonder these parameters can be usefully viewed as tuning parameters. In particular, is it important to take $\sigma^2/(2\lambda)$ large enough for the information of interest to be distinguishable, but not so large that the information is lost in a sea of irrelevant 'noise'?

In the case of a compact Riemannian manifold, a natural choice in the forwards diffusion is to set the drift equal to 0; then the equilibrium distribution is the uniform distribution. But if the information of interest is concentrated in a small portion of the manifold, then perhaps it is helpful to consider alternative diffusions. Consider the unit sphere S^{p-1} in $\mathbb{R}^p$. Two possible choices for the drift along a geodesic from the north pole to the south pole are $b(\theta) = -\lambda \sin \theta$ and $b(\theta) = -\lambda \sin^2 \theta$, where θ is the colatitude. The corresponding equilibrium distributions are the von Mises–Fisher distribution and the Watson distribution (a special case of the Bingham distribution), respectively (e.g. Kent, 1978). The Bingham distribution is mentioned here because there is a classic identification between S_3 and $SO(3)$ through a two-to-one mapping. Hence, any calculation on $SO(3)$ can be recast in terms of a corresponding calculation on S_3. In particular, the Bingham distribution on S_3 can be identified with the matrix von Mises–Fisher distribution on $SO(3)$.

Further, since the isotropic matrix Fisher distribution is similar in behaviour to the Brownian motion distribution of equation (29), and since the isotropic and anisotropic matrix Fisher distributions are easy to simulate efficiently (Kent et al., 2018), they may provide a useful alternative to equation (29) in certain settings. By the way, the wrapping procedure of Section J5 is not very satisfactory in general (except on the circle). In particular, on $SO(3)$ it leads to a density with a singularity at the identity matrix; my colleague Kanti Mardia is giving more details.

Conflict of interest: None declared.

References

Kent J. T. (1978). Time reversible diffusions. *Advances in Applied Probability*, 10(4), 819–835. https://doi.org/10.2307/1426661

Kent J. T., Ganeiber A. M., & Mardia K. V. (2018). A new unified approach for the simulation of a wide class of directional distributions. *Journal of Computational and Graphical Statistics*, 27(2), 291–301. https://doi.org/10.1080/10618600.2017.1390468

Jorge Mateu's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Jorge Mateu

Department of Mathematics, University Jaume I, Castellón, Spain

Address for correspondence: Jorge Mateu, Department of Mathematics, University Jaume I, E-12071 Castellón, Spain. Email: mateu@uji.es

Discussion

The authors are to be congratulated on a valuable and thought-provoking contribution. Deep neural network models have become increasingly prominent across many domains of science, engineering, and industry, finding applications in almost every field. These models have proven particularly valuable when dealing with data exhibiting spatial dependencies (such as images) or temporal dependencies (as in this paper). There is growing interest in combining the ideas and approaches from deep machine learning and neural networks with spatial statistical methods, to capitalize on the expressiveness that deep machine learning models often provide, and/or to approximate intractable or computationally difficult aspects of a well-accepted statistical procedure (Wikle et al., 2023).

I want to emphasize the idea of viewing the change-point detection problem as a classification instead of a testing problem, and here I outline two extensions of detecting change points in time series to spatial (and spatio-temporal) data.

Detecting change points in multivariate settings is usually carried out by analysing all marginals either independently, via univariate methods, or jointly, through multivariate approaches. The former discards any inherent dependencies between different marginals and the latter may suffer from domination/masking among different change points of distinct marginals. Moradi et al. (2023) propose an approach which groups marginals with similar temporal behaviours, and then performs group-wise multivariate change-point detection. The approach groups marginals based on hierarchical clustering using distances which adjust for inherent dependencies. This method significantly enhances the general performance of multivariate change-point detection methods. Adding flexible multi-layer neural networks, as the authors propose in this paper, can surely bring computational gains and generality in detecting the change points.

In a discriminant problem in spatial point patterns, identifying structural differences among observed point patterns from several populations is of interest in several applications. Jalilian and Mateu (2023) use deep convolutional neural networks and employ a Siamese framework to build a discriminant model for distinguishing structural differences between spatial point patterns together with a one-shot learning classification method. In this paper, the authors comment of posing the same strategy when dealing with change points rather than structural characteristics. It would be nice to see how the author's method can be adapted to this context.

I finally would like to call the attention of the authors to the recent paper by Briz and Mateu (2023) in which we use Bayesian inference to detect change points in univariate Hawkes point processes. Adapting the authors method to this other context can open further avenues of interesting research.

Conflicts of interest: None declared.

References

Briz, A., & Mateu, J. (2023). Point process modeling through a mixture of homogeneous and self-exciting processes. *Statistica Neerlandica*. Forthcoming.

Jalilian, A., & Mateu, J. (2023). Assessing similarities between spatial point patterns with a Siamese Neural Network discriminant model. *Advances in Data Analysis and Classification*, *17*(1), 21–42. https://doi.org/10.1007/s11634-021-00485-0

Moradi, M., Cronie, O., Perez-Goya, U., & Mateu, J. (2023). Hierarchical spatio-temporal change-point detection. *The American Statistician*, 77(4), 390–400. https://doi.org/10.1080/00031305.2023.2191670

Wikle, C. K., Mateu, J., & Zammit-Mangion, A. (2023). Deep learning and spatial statistics. *Spatial Statistics*, 57, 100774. https://doi.org/10.1016/j.spasta.2023.100774

Hernando Ombao's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Hernando Ombao

Statistics Program, CEMSE, King Abdullah University of Science and Technology (KAUST), Thuwal, Saudi Arabia

Address for correspondence: Hernando Ombao, Statistics Program, CEMSE, King Abdullah University of Science and Technology (KAUST), Thuwal 23955, Saudi Arabia. Email: hernando.ombao@kaust.edu.sa

I congratulate the authors for their fundamental contribution that demonstrates how the existing tests for change-points can be represented by a neural network. The major consequence is that with sufficient data, the performance of these neural networks should be at par with these existing tests. This formulation is very insightful and inspired me to revisit the previous work on models for non-stationary time series that use time-localized representations such as wavelets, wavelet packets (WP), and the smooth localized complex exponentials (SLEXs). Orthonormal representations are mathematically elegant and they generalize the Cramer representation for stationary time series which use the Fourier basis functions.

One advantage of the WP and SLEX is that they form a library of many bases and thus provide a collection of candidate representations. The best representation is selected data-adaptively to be the one that minimizes the penalized Kullback–Leibler criterion. Due to the localized nature of the basis functions, the best model gives a specific segmentation of the time series data (in the case of SLEX, this is the segmentation of time). Thus, selecting the best model implicitly provides the estimated change-points in the tine series data.

The current implementation limits the choice of the model to have a dyadic segmentation of time (in the case of SLEX) or frequency (in the case of WPs). This dyadic constraint is necessary to take advantage of the best basis algorithm of Coifman and Wickerhauser which requires the magnitude of $O(T)$ operations (where T is the number of time points). The dyadic nature is a primary limitation of these methods because the change-points are constrained to lie on dyadic time points. This is where the current work of Li et al. can provide a major improvement. The dyadic change-points could serve as the initial values of these change-points but could be refined and made more precise by taking advantage of the neural networks.

Conflicts of interest: None declared.

Tamás P. Papp, Paul Fearnhead, and Chris Sherlock's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Tamás P. Papp[1] (iD)**, Paul Fearnhead[2]** (iD) **and Chris Sherlock[2]** (iD)

[1]STOR-i Centre for Doctoral Training, Lancaster University, Lancaster, UK
[2]Department of Mathematics and Statistics, Lancaster University, Lancaster, UK

Address for correspondence: Tamás P. Papp, STOR-i Centre for Doctoral Training, Lancaster University, Lancaster LA1 4YR, UK. Email: t.papp@lancaster.ac.uk

We provide comments only on the read paper "From Denoising Diffusions to Denoising Markov Models" by Benton, Shi, De Bortoli, Deligiannidis and Doucet.

We congratulate the authors on an important contribution to the exciting area of generative models: providing a simple framework for applying the idea of diffusion generative models to data defined on general spaces and for general noising processes.

Diffusion models have shown remarkable empirical performance for several applications, most notably as a way of defining implicit models for images. However it is unclear, to us at least, why these methods are so successful. They typically involve fitting highly parameterized models for the score, and it is unclear why one would expect to be able to find good parameter estimates for such models. Furthermore, such models may be sufficiently flexible to learn the exact denoising process for the samples they are trained on—thereby collapsing the diffusion model to a categorical sampler from the training data. Could the authors provide insight into *why* these methods work? Is there intuition as to what types of data their denoising Markov models will work well for?

Diffusion models are often used when the data are supported on a low-dimensional manifold of the state space. However, the Kullback-Leibler (KL) divergence which these models implicitly minimize can be infinite when the supports of the distributions are disjoint. Can the proposed framework extend to objectives which do not have this issue? Or is the use of KL beneficial, in that it actually helps force the model to sample from close to the manifold?

The proposed framework parameterizes the generative process in terms of a function $\beta(\mathbf{x}; t)$, whose optimal value is the marginal density of the noised data at time t. However, in practice the authors use different parameterizations, mainly based on differences in $\log \beta(\mathbf{x}; t)$. Can the authors give more intuition as to why this is the best choice? And do they think this would be the case in general?

Finally, both parameterizations of the continuous-time Markov chain model revolve around $\log r(\mathbf{x}, \mathbf{y}; t) = \log \beta(\mathbf{x}; t) - \log \beta(\mathbf{y}; t)$. As far as we can tell, the authors do not impose the anti-symmetry $\log r(\mathbf{x}, \mathbf{y}; t) = -\log r(\mathbf{y}, \mathbf{x}; t)$ when fitting r, which seems like some form of relaxation of the optimization problem. Are there advantages in fitting a more general functional form? Even if you fit a general function r, you can apply a post-hoc anti-symmetric correction:

$$\tilde{r}(\mathbf{x}, \mathbf{y}; t) = \left(\frac{r(\mathbf{x}, \mathbf{y}; t)}{r(\mathbf{y}, \mathbf{x}; t)} \right)^{1/2}. \tag{1}$$

Table 1. Image quality metrics for MNIST 14×14 inpainting, with standard errors in brackets

	Original implementation	With correction (1)
Mean-squared error (scaled)	5.49 (± 0.03)	5.49 (± 0.03)
Structural similarity index measure	0.759 (± 0.001)	0.758 (± 0.001)

With the denoising formulation, this does not require additional neural network evaluations. We found that this correction produced almost identical results for the MNIST inpainting example (see Table 1).

Conflict of interest: Paul Fearnhead co-authored the read paper "Automatic Change-Point Detection in Time Series via Deep Learning".

Sam Power's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Sam Power [ORCID]

School of Mathematics, University of Bristol, Bristol, UK

Address for correspondence: Sam Power, Fry Building, Woodland Road, BS8 1UG, Bristol, UK. Email: sam.power@bristol.ac.uk

I thank the authors for a compelling paper, which describes methodology with the potential for very wide applicability. The work approaches the task of extending denoising diffusion models to general spaces by observing that even quite exotic state spaces can still accommodate convenient Markovian evolutions. A recent note of Montanari (2023) has instead obtained generalizations by focusing on indirect 'observation processes' y of the signal x, which could be taken to live on a space of one's own choice (potentially entirely unrelated to the domain of x). Could the authors comment on the advantages which they perceive of the 'Markov perspective' relative to the 'observation perspective'?

Conflict of interest: None declared.

Reference

Montanari, A. (2023). *Sampling, diffusions, and stochastic localization*, arXiv, arXiv:2305.10690, preprint: not peer reviewed.

Johannes Schmidt-Hieber's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Johannes Schmidt-Hieber [ORCID]

Department of Applied Mathematics, University of Twente, Enschede, The Netherlands

Address for correspondence: Johannes Schmidt-Hieber, University of Twente, Drienerlolaan 5, Enschede, NB 7522, The Netherlands. Email: a.j.schmidt-hieber@utwente.nl

I would like to congratulate the authors to this important contribution linking change-point detection and machine learning. My first comment is related to convolutional neural networks (CNNs) that are considered in Section 6. Empirical studies for image classification have shown that in the first hidden layer, CNNs learn, among other features, to detect edges in various directions. This can be viewed as change-point detection in two dimensions. For one change point, the article uses that the cumulative sum (CUSUM) transformation $\mathcal{C}(\mathbf{x}) = (\mathbf{v}_1^\top \mathbf{x}, \ldots, \mathbf{v}_{n-1}^\top \mathbf{x})^\top$ can be realized by the first layer in a neural network with $n - 1$ units in the hidden layer and input vector $\mathbf{x}$. Since

$$\mathbf{v}_i = \big(\underbrace{r_i, \ldots, r_i}_{i \text{ times}}, \underbrace{-r_{n-i}, \ldots, -r_{n-i}}_{n-i \text{ times}} \big)$$

with $r_i := \sqrt{\frac{n-i}{ni}}$, all weight vectors $\mathbf{v}_1, \ldots, \mathbf{v}_{n-1}$ have a similar structure. If the r_i would not depend on i, the CUSUM transformation could even be realized by a feature map in a CNN. But always a small discrepancy remains between the CUSUM approach and feature learning in CNNs. An interesting follow-up question is whether one could derive an analogue of Theorem 4.2 for CNNs instead of fully connected neural networks. To answer this question, the right setting might be (as in Section 6) to assume that there are several change points present and that any two change points could be close. This requires a more local approach and CNNs might then be close to optimal.

My second comment relates to depth. While shallow networks are sufficient for the change-point detection problem considered here, one can wonder what type of relevant higher order structures become learnable by adding layers. This question might be even more relevant for edge detection in the two-dimensional case. It seems conceivable that adding another convolutional layer allows to learn patterns within the detected edges. Consequently, a neural network with two hidden layers could potentially recognize far more complex shapes.

Conflict of interests: None declared.

Frederic Schoenberg and Weng Kee Wong's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Frederic Schoenberg[1] and Weng Kee Wong[2] (iD)

[1]Department of Statistics, University of California at Los Angeles, Los Angeles, USA
[2]Department of Biostatistics, University of California at Los Angeles, Los Angeles, USA

Address for correspondence: Weng Kee Wong, Department of Biostatistics, University of California at Los Angeles, 650 Charles E. Young Drive South, Los Angeles, CA 90095-1772, USA. Email: wkwong@g.ucla.edu

We congratulate the authors for an interesting article. Below are our comments:

1. An important task moving forward will be to investigate under what conditions such algorithms are sub-optimal. For example, certainly if there is a simple parametric form and it is known, correctly, then an estimate taking advantage of this information should be expected to outperform a simple neural network algorithm that does not use this information.
2. The proof that the CUSUM-based statistic corresponds to a neural network estimate is instructive and very nice, but this alone does not ensure that in practice, a neural network is estimated that matches the ideal estimate, even in simple cases.
3. When estimating a change-point in a practical setting, one gets a sense that the problem ultimately boils down typically to whether the values in some time interval are sufficiently different from values in the previous time interval. I wonder if using complex models such as neural networks obscures this somewhat.
4. There are really several different change-point problems, even within the context of estimating a single change-point, rather than multiple change-points. One problem is to pinpoint the time of a change-point, given knowledge that one has occurred in the past. Another is to evaluate in real time whether a change-point seems to have just occurred. A third is to evaluate, retrospectively, whether a change-point seems to have occurred in the past, and perhaps also jointly to pinpoint its timing. This third task is the focus of the current paper. The second task, however, is often to be performed in real time, and having a fully automatic method for accomplishing this task is typically considerably more valuable for this second problem than the others. It seems plausible that neural network algorithms might be more desirable for this second problem of real-time evaluation of whether a change-point has recently occurred, rather than the more retrospective analytical problems of identifying an optimal change-point given knowledge that one has occurred, or assessing after the fact whether or not a change-point has occurred at some time in the past.
5. It would be interesting to see if similar results can be obtained for point processes. For disease epidemics, for example, one often obtains point process data on incidence times and it is of significant interest to determine if there is a change in the process that could indicate a significant mutation of the disease.

Conflicts of interest: None declared.

Yingnian Wu and Weng Kee Wong's contribution to the Discussion of 'the Discussion Meeting on Probabilistic and statistical aspects of machine learning'

Ying Nian Wu[1] and Weng Kee Wong[2] (iD)

[1]Department of Statistics, University of California at Los Angeles, California, USA
[2]Department of Biostatistics, University of California at Los Angeles, California, USA

Address for correspondence: Weng Kee Wong, Department of Biostatistics, University of California at Los Angeles, 650 Charles E Young Dr S, Los Angeles, CA 90095, USA. Email: wkwong@ucla.edu

We thank the authors for the interesting and timely article. Our commentaries are as follows:

From equilibrium Langevin dynamics to non-equilibrium ordinary and stochastic differential equations. We wish to add a little background explanation, with no claim of originality, to this deep and elegant paper that greatly expands the scope of diffusion/score-based models. The diffusion-based model can be viewed as a refined variational auto-encoder and a non-equilibrium Langevin sampler. While the variational auto-encoder perspective seems more fundamental because it directly targets the log-likelihood through the variational lower bound, the non-equilibrium Langevin perspective is also revealing.

Specifically, for a target density $p(x)$, the Langevin dynamics for sampling $p(x)$ is

$$x_{t+\Delta t} = x_t + \frac{\Delta t}{2}\nabla_x \log p(x_t) + \sqrt{\Delta t}\epsilon_t,$$

where Δt is discretized time step, and $\epsilon_t \sim \mathcal{N}(0, I)$ independently over discretized t (more rigorous treatment involves taking the limit $\Delta t \to 0$). The Langevin dynamics consists of a deterministic gradient ascent term and a stochastic noise injection term. It is an equilibrium sampler in that if $x_t \sim p(x)$, then $x_{t+\Delta t} \sim p(x)$. However, for Markov chain Monte Carlo sampling, we have to start from a simple initial noise distribution and iterate the Langevin dynamics for infinite time in order to converge to $p(x)$.

But the familiar Langevin dynamics tells us a simple yet profound fact if we separate the deterministic gradient ascent step and the stochastic noise injection step. Suppose $x \sim p(x)$, let $x_- = x + \frac{\Delta t}{2}\nabla_x \log p(x) \sim p_-(x)$ by gradient ascent. Then $x_- + \sqrt{\Delta t}\epsilon \sim p(x)$ after noise injection, where $\epsilon \sim \mathcal{N}(0, I)$. Put it another way, if $x_- \sim p_-(x)$, and $x_- + \sqrt{\Delta t}\epsilon \sim p(x)$ after noise injection, then $x + \frac{\Delta t}{2}\nabla_x \log p(x) \sim p_-(x)$. That is, the deterministic gradient ascent step reverses the noise injection step as far as the marginal distribution is concerned.

This suggests that if we start from the data distribution $x_0 \sim p_0(x)$, and iterates the noise injection step $x_{t+\Delta t} \sim x_t + \sqrt{\Delta t}\epsilon_t$, then we can reverse it by the gradient ascent step $\tilde{x}_{t-\Delta t} = \tilde{x}_t + \frac{\Delta t}{2}\nabla_x \log p_t(\tilde{x}_t)$, where p_t is the density of x_t, and we use $\tilde{x}$ notation for the samples along the reverse trajectory, where $\tilde{x}_t$ has the same marginal distribution as x_t. This leads to denoising by ordinary differential equation. If we double the step size of gradient ascent, then $\tilde{x}_t + \Delta t\nabla_x \log p_t(\tilde{x}_t)$ amounts to $\tilde{x}_{t-2\Delta t}$, and we need to inject noise $\sqrt{\Delta t}\tilde{\epsilon}_t$ to get back to the marginal distribution at $t - \Delta t$. This leads to denoising by stochastic differential equation: $\tilde{x}_{t-\Delta t} = \tilde{x}_t + \Delta t\nabla_x \log p_t(\tilde{x}_t) + \sqrt{\Delta t}\tilde{\epsilon}_t$. Both ordinary and stochastic differential equations are finite-time non-equilibrium samplers.

Conflict of interests: None declared.

The authors replied later, in writing, as follows:

Authors' reply to the Discussion of 'Automatic change-point detection in time series via deep learning' at the Discussion Meeting on 'Probabilistic and statistical aspects of machine learning'

Jie Li[1] [iD]**, Paul Fearnhead[2]** [iD]**, Piotr Fryzlewicz[1] and Tengyao Wang[1]**

[1]Department of Statistics, London School of Economics and Political Science, Columbia House, Houghton Street, London, WC2A 2AE, UK
[2]Department of Mathematics and Statistics, Lancaster University, Lancaster, LA1 4YF, UK

Address for correspondence: Jie Li, Department of Statistics, London School of Economics and Political Science, Columbia House, Houghton Street, London, WC2A 2AE, UK. Email: j.li196@lse.ac.uk

We would like to thank the proposer, seconder, and all discussants for their time in reading our article and their thought-provoking comments. We are glad to find a broad consensus that neural-network-based approach offers a flexible framework for automatic change-point analysis. There are a number of common themes to the comments, and we have therefore structured our response around the topics of the theory, training, the importance of standardization and possible extensions, before addressing some of the remaining individual comments.

Theory. Both Wilkinson and Zhang compare the theoretical bound on the generalization error in Theorem 4.2 with our empirical results in Figure 2 of main text, Figures S2 and S3 of supplement. Our experience has been that the empirical generalization error has been substantially lower than the theoretical bounds suggest—with good performance of our fitted neural network with training sample sizes that are orders of magnitude smaller than one may expect given the number of parameters within the neural network. Related to this is that how the bound on the generalization error depends on e.g. the number of layers, is not particularly informative about how these factors affect the error in practice. We agree with both Wilkinson and Zhang that there is a need for a new theoretical framework for bounding the generalization error of neural networks that is more meaningful in practice.

We thank Zhang for pointing out how our theoretical analysis can be extended to more general data generating mechanisms, including heavier-than-Gaussian noise distributions and data with weak temporal correlation (a concern of Hong et al., 2024). Indeed, as Zhang comments, the same procedure still works in such settings and the current proof will go through, with minor modifications to the choice of λ in Corollary 4.1. As λ is adaptively chosen by the neural network (which is one of the main attractions of our procedure), the results in Theorems 4.2 and 4.3 will be essentially unchanged. In a similar vein, Schmidt-Hieber mentions that our theory could be modified to prove local change-point detection error rates using convolutional neural networks (CNN). Indeed, by combining existing results on moving sum (MOSUM) (Eichinger & Kirch, 2018) together with Vapnik-Chervonenkis (VC) dimension results of CNN, we could arrive at a similar result to Theorems 4.2 and 4.3 in the paper.

Gavioli-Akilagun points out that the current architecture cannot directly exactly represent the likelihood-ratio test statistic for detection of piecewise affine changes. We agree with this observation. However, by including squared observations as inputs, or if we include squaring in the set of nonlinearities permitted by the neural network, we can directly express the likelihood-ratio test as a function in the neural network class (VC-dimension results concerning neural networks with piecewise polynomial activation functions are given in Theorem 7 of Bartlett et al., 2019). Moreover, by including multiple layers within our architecture, the neural network can accurately approximate the square function. This is related to the nice results of Gavioli-Akilagun that show

test statistics based on linear functions, which can be represented simply by a neural network, can have statistical performance comparably to those based on the likelihood-ratio test.

Regarding Zhang's observation that the first term in Theorem 4.3 does not decrease as training sample size N increases, our intuition is that the first term represents the Bayes risk of the classification task on the test set, which is achieved by the cumulative sum based classifier. Therefore, it will not be affected by increasing training sample size and only depends on the test sample.

Neural network architecture. Both Wilkinson and Nemeth raise the question of how to choose a suitable neural network architecture for different settings. This is a well-studied problem in machine learning. A few possible Neural Architecture Search approaches are mentioned in Paaß and Giesselbach (2023, Section 2.4.3) (also discussed in Section 6 of the main text). Nemeth also asks the question of whether it is easier to choose the neural network architecture than to choose stochastic models to represent the data. Given these Neural Architecture Search methods mentioned above, we believe that the former is at least a more structured problem that can be solved algorithmically.

Schmidt-Hieber discusses the effect of the depth of the network on its ability to detect various structures in the signal. Indeed, we agree that in general the less we know about the data generating mechanism, the more layers we need in the architecture.

Training. Cribben and Anastasiou raise a number of important practical considerations with training. First, as our approach requires labelled data, there is the challenge of obtaining such data. We agree that for manually labelled data, the labelling of changes is subjective, and this means that our method will only aim to replicate the manual classification. If we use simulation, then we can avoid this, but at the expense of needing to model what changes would look like. They also point out that often changes are rare—so training data may be imbalanced, and we would also want to account for this imbalance when detecting new changes. If we believe the frequency of changes will be different in the training than in the test data, or if we wish to account differently for different types of errors (false detection versus missing a true change), this is possible by including different weights for each error when training.

Both Bodenham and Adams, and Hong et al., ask how our approach could be applied to a single long time series, for example from finance. First, our method requires labelled training data, so we would require part of the time series to have labelled changes, or known to have no changes. In this case, we can divide such historical data into time series of a fixed length, leading to a set of labelled training data. One can then fit a neural network classifier to this data and use the fitted neural network to classify windows of new data using the same moving window idea as in Section 6.

When not enough training data is available, we proposed to simulate artificial data to train our neural network. Wilkinson points out that this has a close link to the simulation-based inference. We agree that it is worth investigating the links with this literature further.

Standardization. We agree with Chen and Chen that the simple neural network classifier is not automatically invariant to the shifting and scaling because it may not have learned an exactly invariant statistic.

Differences in the results for standardized versus nonstandardized data show that the algorithm (not unexpectedly) learns to perform classification for the problem at hand, whether or not it aligns with the analyst's perceived best way of distinguishing the two classes. More specifically, the algorithm's task is to solve a binary classification problem, distinguishing between two groups of sequences. While the analyst may suspect that it is the presence of the change point that is the main feature separating the two classes, the learning algorithm may take a different view given the training data. For example, in the nonstandardized case, if all the input data starts with $\mu_L = 0$, what the analyst regards as a change-point problem the algorithm may construe as a testing problem of departure from a zero mean.

Furthermore, as pointed out by Wilkinson, the min–max scaling used in our algorithm may not be appropriate for very heavy-tailed data. In such contexts, scaling by empirical quantiles other than 0 and 1 would be more appropriate.

In many applications, one would also like the test to be invariant to the reversal of the time direction. Hence, Chen and Chen's proposal of adding reversed sequence $X_n, \ldots, X_1$ into the training data, which has the additional benefit of enlarging the sample size, makes sense.

Extensions. As Bodenham and Adams point out, often one is interested in localizing changes, rather than just detecting them. As a first work in the area of using neural networks to automatically construct change-point detectors, we deliberately focused on the problem of detection rather than localization. However frameworks similar to that of MOSUM (Eichinger & Kirch, 2018), where we apply a detector to different windows of data, can use a change-point detector to also estimate the location of any changes. We used this idea, i.e. Algorithm 1, in the analysis of the Human Activity Sensing Consortium data in Section 6. However, we believe that there may be more attractive ways of extending our idea to the problem of change-point localization.

A number of discussants (Anastasiou and Cribben, Schoenberg and Wong, Bodenham and Adams) ask whether our method can be applied in an online setting. This is possible provided that we have a pretrained neural network that can classify data of a given window length, say h. When we receive a new observation, we can apply the neural network classifier to the most recent h time points. Computationally this is possible in an online setting, as once trained, the cost of running the classifier is fixed. There are challenges in terms of how to tune the classifier so that the resulting online change-point detector has an appropriate average run length. Also, how to extend this idea so that we also update the classifier online as we get new data, is an interesting open question.

Schoenberg and Wong, Anastasiou and Cribben, Mateu, van Lieshout and Lu ask whether our ideas could be extended to multivariate data, and in particular spatial data or point process data. We are interested to hear about the recent developments in this area and look forward to seeing future works in this direction. One challenge of dealing with point process data is that the number of points is random and cannot be easily interpreted as input of a neural network. As a first order approximation, we could bin the data into a (multivariate) time series of counts and a similar method to the one proposed in our article could then be applied.

Other comments. We are interested to read about other research at the interface between neural networks and change point and related areas in statistics. This includes the using change-point detection to improve the fitting of deep neural network models to time-series data (Jungbluth & Lederer, 2023) and the possibility of using ideas from our article to improve existing change-point detection methods as suggested by Ombao. Schmidt-Hieber also points out the possibility of using a convolutional neural network-based approach for detecting local change points or two-dimensional edge detection. Finally, we agree with MacKenzie that one disadvantage with automated procedures like the one in our article is that the final test statistic for a change is hard to interpret. We welcome work on improving interpretability of artificial intelligence (AI) and believe ideas in this area will be important as AI methods are increasingly using within statistics.

Conflicts of interest: We have no conflict of interest to disclose.

References

Bartlett P. L., Harvey N., Liaw C., & Mehrabian A. (2019). Nearly-tight VC-dimension and pseudodimension bounds for piecewise linear neural networks. *Journal of Machine Learning Research*, 20(63), 1–17. https://doi.org/10.48550/arXiv.1703.02930

Eichinger B., & Kirch C. (2018). A MOSUM procedure for the estimation of multiple random change points. *Bernoulli*, 24(1), 526–564. https://doi.org/10.3150/16-BEJ887

Hong Y., Linton O., McCabe B., Sun J., & Wang S. (2024). Kolmogorov–Smirnov type testing for structural breaks: A new adjusted-range based self-normalization approach. *Journal of Econometrics*, 238(2), 105603. https://doi.org/10.1016/j.jeconom.2023.105603

Jungbluth A., & Lederer J. (2023). 'The DeepCAR method: Forecasting time-series data that have change points', arXiv, arXiv:2302.11241, preprint: not peer reviewed.

Paaß G., & Giesselbach S. (2023). *Foundation models for natural language processing: Pre-trained language models integrating media*. Artificial intelligence: Foundations, theory, and algorithms. Springer International Publishing.

Authors' reply to the Discussion of 'From denoising diffusions to denoising Markov models' at the Discussion Meeting on 'Probabilistic and statistical aspects of machine learning'

Joe Benton[1] (ORCID), **Yuyang Shi[1], Valentin De Bortoli[2], George Deligiannidis[1] and Arnaud Doucet[1]**

[1]Department of Statistics, University of Oxford, Oxford, UK
[2]École normale superiéure, Paris, France

Address for correspondence: Joe Benton, Department of Statistics, University of Oxford, 24-29 St Giles', Oxford, OX1 3LB, UK.
Email: benton@stats.ox.ac.uk

1 Response

We would like to thank all the discussants for their contributions and comments on our work.

1.1 Inductive biases of DMMs

The question of why and on what class of distributions denoising Markov models (DMMs) work well in practice remains solidly open, as was highlighted by several discussants. Nevertheless, we may offer several hypotheses supported by recent investigations. Firstly, for a denoising diffusion in $\mathbb{R}^d$ approximating a data distribution supported on a submanifold, for small times t the score function $\nabla \log q_t(\mathbf{x})$ points locally in the direction towards the manifold and explodes in magnitude. This behaviour is induced by the implicit Kullback-Leibler (KL) objective we minimize and forces the model to sample from close to the manifold, as suggested by Papp, Fearnhead and Sherlock, helping the model to recover the underlying data manifold (Stanczuk et al., 2023; Wenliang & Moran, 2023). We hypothesize that something similar may happen for DMMs more generally, whereby they pick up the geometry of the data distribution through observing local changes in $q_t(\mathbf{x})$ for small times. Other work has identified conditions under which diffusion models on $\mathbb{R}^d$ will accurately reconstruct the data manifold (Pidstrigach, 2022), but diffusion models in practice operate far from the regime discussed therein.

In any case, a generative model which perfectly memorized the empirical distribution would be useless, as it would not generalize beyond the training data. So, to understand the success of DMMs, we must also understand how the inductive bias of the score approximation network allows them to interpolate between observed samples. It would appear that diffusion models have powerful inductive biases encouraging this interpolation, since it has been observed that diffusion models trained with the same architecture on disjoint training datasets often end up learning to generate very similar images (Kadkhodaie et al., 2023). Remarkably, it has ever been observed experimentally by Zhang et al. (2023) that 'when starting with the same initial noise input and sampling with a deterministic solver, diffusion models tend to produce nearly identical output content. This consistency holds true regardless of the choices of model architectures and training procedures'.

Work bounding the error of diffusion models on $\mathbb{R}^d$ has decomposed the difference between the data distribution and the learned approximation into three terms, one arising from the approximation of the marginal q_T of the forward process at time T with the reference distribution q_{ref}, one arising from the time-discretization of the reverse stochastic differential equation (SDE), and one arising from the error of the score approximation (Chen et al., 2023c; Chen et al., 2023a). Empirically, we find that the last of these dominates, suggesting that the observed generalization is coming mostly from the manner in which the score is approximated.

There are three factors affecting the approximation learned for the score, which were highlighted in Wang's response. The first is our choice of the ELBO training objective, which we expect generally imparts a favourable bias—on $\mathbb{R}^d$ it reduces to the L^2 score matching objective, which has an elegant and intuitive interpretation in terms of denoising score matching with multiple noise levels (Song & Ermon, 2019; Vincent, 2011). The second is the use of gradient-based optimizers such as stochastic gradient descent or Adam to learn the parameters of our approximation, which as Wang highlights often finds solutions which generalize well. The third is the architecture of the neural network used to parameterize the score approximation. A typical choice of architecture is the UNet, which seems empirically to perform well (Ho et al., 2020). There have been some initial investigations into the appropriateness of the inductive bias of UNets for diffusion models [see, for example Williams et al., 2023 or the experiments provided by Kadkhodaie et al. (2023) and Gilliot, Andrieu, Lee, Liu, and Whitehouse]. However, our understanding of the relevance of the neural network architecture is far from complete, and the questions of which biases are induced by the common choices of architectures and how to encode a given set of biases through an appropriate such choice remains understudied.

In general, much of the existing literature on the inductive biases of diffusion models applies only to the case of models on $\mathbb{R}^d$. Understanding the bias of arbitrary DMMs will inevitably be a much greater challenge.

1.2 Inference techniques

One of the main practical considerations when implementing DMMs is how to simulate the forward and reverse processes. While we focused mostly on simple first-order methods in our paper, finding improved inference techniques may greatly improve the performance of DMMs in practice, since the number of steps required to simulate the reverse process dictates the number of neural function evaluations required, which is typically the limiting factor in performance. A wide range of techniques have been developed for speeding up diffusion models in $\mathbb{R}^d$, including the probability flow ordinary differential equation (probability flow ODE, or PF ODE) (Song, Sohl-Dickstein, et al., 2021), consistency models (Song et al., 2023), distillation (Salimans & Ho, 2022), using higher order ODE simulation methods (Karras et al., 2022), or using non-Markovian noising processes such as in the denoising diffusion implicit model (DDIM) or flow matching methods (Lipman et al., 2023; Liu et al., 2023; Song, Meng, et al., 2021).

Unfortunately, most of these techniques do not immediately transfer to the more general DMM setting. Nevertheless, we are hopeful that specific techniques can speed up inference for particular classes of DMMs. For example, the tau-leaping method developed by Campbell et al. (2022) and which we use in Section 6.2 can be viewed as one such method. Other possibilities may include extending the probability flow ODE to new state spaces—indeed, De Bortoli et al. (2022) show how the PF ODE can be extended to the Riemannian manifold setting—though we are not currently aware of a straightforward and computationally tractable generalization of the PF ODE to arbitrary DMMs. Alternatively, we could find higher order simulation methods or adaptive time-stepping procedures for other classes of Markov processes, as suggested by Wilkinson, which would mitigate the need for time-rescaling.

1.3 Choosing the forward process

In response to Kent's question regarding the choice of hyperparameters for the Ornstein-Uhlenbeck (OU) forward process, we note that the forward process should be run until (approximate) convergence—that is, until the information in the original distribution is essentially completely destroyed. Assuming we do this, any choices of λ and σ will in fact lead to an equivalent diffusion model, up to linear rescalings of time and space. Therefore, there is little theoretical difference to be made by tuning the parameters λ and σ, beyond the inductive bias that particular choices impart when learning the score approximation.

In practice, it is found that it is easiest to learn the score function if the data are normalized to take values in some standard interval. For example, for data on $\mathbb{R}^d$, we typically rescale the data to lie in the interval $[-1, 1]$, or have identity covariance, and then choose our forward process to have the standard Gaussian as its invariant distribution (Ho et al., 2020). For a similar

reason, we also typically apply a time-rescaling function $\beta(t)$ to make the score easier to learn near $t = 0$, as described in Song, Sohl-Dickstein, et al. (2021).

1.4 Parameterization of the reverse process

There are several reasons why we tend to prefer parameterizing the reverse process in terms of a logarithmic gradient of $\beta(\mathbf{x}, t)$ rather than via $\beta(\mathbf{x}, t)$ directly. The first is that we expect the $\nabla \log$ operator to smooth the target substantially. At the optimum, $\beta(\mathbf{x}, t)$ is equal to the marginal $q_t(\mathbf{x})$, which may vary over many orders of magnitude when t is close to 0. Accordingly, $\log \beta(\mathbf{x}, t)$ tends to be a much more stable training target. Second, we find that the objective $\mathcal{I}_{\text{ISM}}$ and the reverse process are often more naturally parameterized by $\nabla \log \beta(\mathbf{x}, t)$ or its equivalent, as can be seen for example in the real diffusion case in Appendix J.1. If it is $\nabla \log \beta(\mathbf{x}, t)$ which features in both our training objective and our description of the reverse process, we believe that it makes most sense to target this quantity directly. Finally, we find that this choice leads empirically to more stable training, validating our design decision. We expect this to be the case in most (though perhaps not all) applications of DMMs.

Concerning our decision not to enforce symmetry constraints on $\log r(\mathbf{x}, \mathbf{y}; t)$ in the continuous-time Markov chain (CTMC) example, this was mostly for practical convenience, since it is simpler to parameterize the unconstrained function. In addition, we did not find that the performance of our models were noticably negatively affected by this lack of symmetry, as Papp, Fearnhead, and Sherlock themselves note. We speculate that this may be a particular instance of a common phenomenon in machine learning whereby discarding certain domain-specific knowledge and optimizing over an unconstrained state space turns out to perform just as well, if not better, than a more heavily customized model.

1.5 Novel applications of DMMs

One of our main motivations for introducing DMMs was to make it possible to apply the powerful denoising diffusion modelling technique to a much broader set of problems. While we had limited space to address such applications in our main work, we were extremely gratified by the many potential real-world applications of DMMs that discussants highlighted, including in theoretical neurobiology (Cialfi), learning highly concentrated directional distributions (Kent), and creating synthetic datasets for statistical disclosure control (Jackson). Applications to spaces with mixed discrete and continuous components, as suggested by Wilkinson, are common in protein generation; see e.g. Morehead et al. (2024).

1.6 Application to *g*-and-*k* distribution

We expect DMMs to be most useful for very high-dimensional or multi-modal distributions. The primary purpose of providing the *g*-and-*k* example in Section 6.1 was to demonstrate that DMMs can also be competitive in standard Bayesian settings, and to evaluate their performance in a case where the ground truth was known and we could perform a comparison to other methods. We found that DMMs were at least as effective as other simulation-based inference (SBI) methods, though we agree with Nemeth and Wilkinson that the large computational cost of DMMs is a drawback which must be weighed against their benefits when considering whether to apply DMMs in such situations. This computational cost is likely a substantial part of the reason why DMMs are not yet used more widely for SBI.

Nemeth notes that the neural networks we use for inference in the *g*-and-*k* example are vastly overparameterized compared to both the observed dataset and the underlying number of parameters. However, we tend to view such overparameterization as an asset rather than a hindrance—in the machine learning literature, it is commonly found that such overparameterization provides a favourable inductive bias, leading us to learn a score approximation that generalizes well. Regarding how many parameters are required to have a high degree of statistical accuracy, Chen et al. (2023b) provide bounds on the size of a neural network required for a given accuracy of score approximation in the case where the data distribution is supported on a linear submanifold.

Of course, where DMMs really come into their own is on problems that are much more complicated than the simple *g*-and-*k* distribution, such as the image inpainting example discussed later. For such high-dimensional problems, classical SBI techniques are infeasible and denoising-based methods really shine.

1.7 Singularities on SO(3)

As Mardia highlights, many methods for defining or approximating distributions on SO(3) may run into problems arising from using singular charts for the manifold. We take this opportunity to clarify how our DMMs avoid this problem by respecting the manifold structure of SO(3). Mardia rightly points out that we use a wrapping procedure to construct the synthetic data distribution on which we test our DMMs, and that this wrapping procedure could cause our test distributions to be non-smooth. However, this simply means that the distributions we test our method by approximating may be singular—it does not reflect a limitation of our DMM method.

Conversely, we would like our learning procedure to be smooth and respect the manifold structure of SO(3), and indeed this was a significant motivation for introducing DMMs. When we apply DMMs on SO(3), we use a noising process—the Brownian diffusion—which respects the manifold structure, and we parameterize the reverse process in terms of a function $\{s_\theta^i(\mathbf{x}, t)\}_{i=1}^3$, which is represented as a three-dimensional vector constrained to lie in the relevant tangent plane (for details, see Section 6.3 and Appendix J.5 of the paper). Accordingly, our parameterization of the reverse process is free of singularities and our learning procedure respects the structure and symmetry of SO(3).

Our decision to use $M = 16$ components for the synthetic test distributions was a somewhat arbitrary design choice made to strike a good balance between significant multi-modality and ease of visual representation. The values of σ_m are drawn randomly with mean 0.1.

Regarding how we assess our methods, we chose to compare to the ground truth for inference problems where it was available in order to verify that our method produced sensible predictions. However, in many cases such as image generation, perhaps the most important metrics will concern the perceptual quality of samples. Hence, for these problems we focused on providing a range of samples from our models, and demonstrating that the samples we get appear to be diverse and representative of the true image manifold. To further assess the performance in the absence of a ground truth, we may also consider using metrics such as the maximum mean discrepancy (Gretton et al., 2012) to assess the distance between the distribution of the generated samples and the training data.

1.8 Relationship to stochastic localization

Power notes that recent work of Montanari (2023) has shown that real diffusion models are equivalent to stochastic localization sampling processes. This correspondence gives new powerful tools for studying diffusion models and their convergence properties—indeed, in separate work we have recently used this result to provide improved convergence bounds for real diffusion models (Benton et al., 2023).

Nevertheless, the denoising perspective on diffusion models retains certain advantages. For example, it suggests natural generalizations of diffusion models using non-Markovian forward processes, as in Song, Meng, et al. (2021) or Lipman et al. (2023). In addition, to make sampling from the observation process tractable, Montanari (2023) rely on a result which essentially categorizes the time-reversal of an SDE as another SDE (their Proposition 1.1). In order to apply the observation perspective on a general state space, we would need an equivalent of this result that held for arbitrary Markov processes. Perhaps this may be done, but our DMM work from our Markovian perspective circumvents the need for such a result—and indeed may be viewed from one perspective as deriving a result of this form, since if we assume the reverse process is learned perfectly, then it will be equal to the time-reversal of the forward process, giving a way to identify this time-reversal in the general case.

Conflict of interests: None declared.

References

Benton J., De Bortoli V., Doucet A., & Deligiannidis G. (2023). 'Linear convergence bounds for diffusion models via stochastic localization', arXiv, arXiv:2308.03686, preprint: not peer reviewed.

Campbell A., Benton J., De Bortoli V., Rainforth T., Deligiannidis G., & Doucet A. (2022). A continuous time framework for discrete denoising models. *Advances in Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2205.14987

Chen S., Chewi S., Li J., Li Y., Salim A., & Zhang A. R. (2023a). Sampling is as easy as learning the score: Theory for diffusion models with minimal data assumptions. *International Conference on Learning Representations*, https://doi.org/10.48550/arXiv.2209.11215

Chen M., Huang K., Zhao T., & Wang M. (2023b). 'Score approximation, estimation and distribution recovery of diffusion models on low-dimensional data', arXiv, arXiv:2302.07194, preprint: not peer reviewed.

Chen H., Lee H., & Lu J. (2023c). Improved analysis of score-based generative modeling: User-friendly bounds under minimal smoothness assumptions. *International Conference on Machine Learning*. https://doi.org/10.48550/arXiv.2211.01916

De Bortoli V., Mathieu E., Hutchinson M., Thornton J., Whye Teh Y., & Doucet A. (2022). Riemannian score-based generative modeling. *Advances in Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2202.02763

Gretton A., Borgwardt K. M., Rasch M. J., Schölkopf B., & Smola A. (2012). A kernel two-sample test. *Journal of Machine Learning Research*, 13(1), 723–773.

Ho J., Jain A., & Abbeel P. (2020). Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2006.11239

Kadkhodaie Z., Guth F., Simoncelli E. P., & Mallat S. (2023). 'Generalization in diffusion models arises from geometry-adaptive harmonic representation', arXiv, arXiv:2310.02557, preprint: not peer reviewed.

Karras T., Aittala M., Aila T., & Laine S. (2022). Elucidating the design space of diffusion-based generative models. *Advances in Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2206.00364

Lipman Y., Chen R. T. Q., Ben-Hamu H., Nickel M., & Le M. (2023). Flow matching for generative modeling. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2210.02747

Liu X., Gong C., & Liu Q. (2023). Flow straight and fast: Learning to generate and transfer data with rectified flow. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2209.03003

Montanari A. (2023). 'Sampling, diffusions, and stochastic localization', arXiv, arXiv:2305.10690, preprint: not peer reviewed.

Morehead A., Ruffolo J. A., Bhatnagar A., Madani A., & Bio P. (2024). 'Towards joint sequence-structure generation of nucleic acid and protein complexes with SE(3)-discrete diffusion', arXiv, arXiv:2401.06151, preprint: not peer reviewed.

Pidstrigach J. (2022). Score-based generative models detect manifolds. *Advances in Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2206.01018

Salimans T., & Ho J. (2022). Progressive distillation for fast sampling of diffusion models. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2202.00512

Song J., Meng C., & Ermon S. (2021). Denoising diffusion implicit models. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2010.02502

Song Y., Dhariwal P., Chen M., & Sutskever I. (2023). Consistency models. *International Conference on Machine Learning*. https://doi.org/10.48550/arXiv.2303.01469

Song Y., & Ermon S. (2019). Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.1907.05600

Song Y., Sohl-Dickstein J., Kingma D. P., Kumar A., Ermon S., & Poole B. (2021). Score-based generative modeling through stochastic differential equations. *International Conference on Learning Representations*. https://doi.org/10.48550/arXiv.2011.13456

Stanczuk J., Batzolis G., Deveney T., & Schönlieb C.-B. (2023). 'Your diffusion model secretly knows the dimension of the data manifold', arXiv, arXiv:2212.12611, preprint: not peer reviewed.

Vincent P. (2011). A connection between score matching and denoising autoencoders. *Neural Computation*, 23(7), 1661–1674. https://doi.org/10.1162/NECO_a_00142

Wenliang L. K., & Moran B. (2023). 'Score-based generative models learn manifold-like structures with constrained mixing', arXiv, arXiv:2311.09952, preprint: not peer reviewed.

Williams C., Falck F., Deligiannidis G., Holmes C., Doucet A., & Syed S. (2023). A unified framework for U-Net design and analysis. *Advances in Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2305.19638

Zhang H., Zhou J., Lu Y., Guo M., Shen L., & Qu Q. (2023). 'The emergence of reproducibility and consistency in diffusion models', arXiv, arXiv:2310.05264, preprint: not peer reviewed.