Estimating Transfer Entropy under Long Ranged Dependencies

Abstract

Estimating transfer entropy (TE) between time series is a difficult, yet highly impactful problem in fields such as finance and neuroscience. The well-known nearest neighbor estimator of TE potentially fails if temporal dependencies are *noisy and long ranged*, primarily because it relies on the estimation of *joint entropy* terms in high dimensions, which is a hard problem in itself. Other estimators, such as those based on Copula entropy or conditional mutual information have similar limitations. Leveraging the successes of modern discriminative models that operate in high dimensional (noisy) feature spaces, we express TE as a difference of two *conditional entropy* terms, which we directly estimate from conditional likelihoods computed in-sample from any discriminator trained per maximum likelihood principle. The challenge here is to ensure that the in-sample log likelihood estimates are not overfit to the data. Inspired by recent work on adversarial robustness of deep neural nets, we propose a novel perturbation model based on locality sensitive hash (LSH) functions, which regularizes a discriminative model to have smooth functional outputs within local neighborhoods of the input space. We theoretically prove that our estimator is consistent under some mild regularity conditions, and its variance reduces linearly in sample size. We also demonstrate its superiority w.r.t. state-of-the-art estimators through empirical evaluations on a synthetic as well as real world datasets from the neuroscience and finance domains.

1 INTRODUCTION

Information theory plays a central role in modern machine learning for tasks like clustering, feature selection, representation learning, autoencoding, generative modeling, fairness, and many other (Cover, 1999; Cicalese et al., 2019; Vergara and Estévez, 2014; Kingma and Welling, 2019; Goodfellow et al., 2014; Song et al., 2019; Song and Kingma, 2021). A relatively new concept in information theory is *transfer entropy* (TE), which quantifies the reduction in uncertainty about one time series, given another (Schreiber, 2000) (see Fig. 1). TE is theoretically and practically appealing for various domains, including finance and neuroscience (Barnett et al., 2009; Vicente et al., 2011; Jizba et al., 2012; Ver Steeg and Galstyan, 2012; He and Shang, 2017; Ursino et al., 2020; Restrepo et al., 2020; Sipahi and Porfiri, 2020), although, it is challenging to estimate TE in practice (Lindner et al., 2011; Barnett and Bossomaier, 2012; Zuo et al., 2013; Vicente and Wibral, 2014; Zhu et al., 2015; Wollstadt et al., 2018; Zhang et al., 2019), especially under *long-ranged & noisy temporal dependencies*. With the recent rapid advance of simultaneous high-density recordings of neural activities across multiple brain areas (Siegle et al., 2021; Steinmetz et al., 2021), it is essential to have a scalable and robust model for estimating TE between neural ensembles with the presence of sparsity signal and large noise due to ubiquitous neuron-to-neuron or trial-to-trial variance (Steinmetz et al., 2018; Kass et al., 2018). In finance domain, given the low-signal-to-noise ratio, empirical models would leverage from advances in TE estimation by filtering out very weak explanatory time series (Dimpfl and Peter, 2013; Sandoval, 2014; Kim et al., 2020).

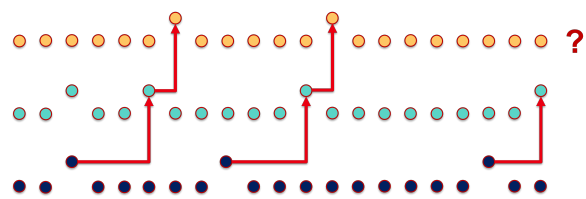


Figure 1: Three different time series are shown in order to illustrate *transfer entropy* (TE). There is a clear pattern of TE from the second time series (green) to the first one (yellow), i.e. predictability of the observations in the first time series given the knowledge of the second one. Similarly, there is TE from the third time series (navy) to the second one, though the dependencies are relatively complex.

A popular technique for estimating TE is based on the k nearest neighbors (kNN) method (Kraskov et al., 2004; Lindner

Submitted to the 38th Conference on Uncertainty in Artificial Intelligence (UAI 2022). **To be used for reviewing only**.

et al., 2011; Lombardi and Pant, 2016; Singh and Póczos, 2016; Lizier, 2014; Montalto et al., 2014; Wollstadt et al., 2018). However, this measures TE indirectly, through joint entropy (Kozachenko and Leonenko (1987)), and becomes problematic when temporal dependencies are noisy. A related approach is to estimate TE via Copula joint entropy or conditional mutual information (Ma, 2019; Zhang et al., 2019), which is also susceptible to dependency based noise.

Noting that TE is represented by two conditional entropy components, in this paper, we propose a discriminative learning approach to the problem. Specifically, we obtain *in-sample* estimates of conditional likelihoods from a discriminative model to estimate the conditional entropy terms directly. This allows us to exploit modern machine learning methods to predict a low dimensional variable, conditioned on a very high dimensional variable, even when many of the dimensions are noisy. Any discriminative model, trained as per the maximum likelihood principle, can obtain in-sample estimates of the conditional log likelihood. Deep neural nets trained with mean squared error as the loss function provide estimates of the conditional log likelihood for free, if errors are assumed to be Gaussian distributed. In our approach, one can also employ any probabilistic regression model (Zhu and Laptev, 2017; Dabney et al., 2018; Rasul et al., 2021) for obtaining the conditional likelihoods, though we advocate for the approach mentioned above which accommodates a large variety of time series forecasters (Oreshkin et al., 2019; Bai et al., 2018). For discrete-valued time series, one can obtain conditional log likelihood from any classifier trained with cross-entropy loss.

It is important to ensure that the in-sample estimates of conditional log likelihood are not overfit to the time series. This is particularly relevant when the estimate is for quantifying if additional information from another time series improves the (in-sample) predictability of a given time series. Intuitively, a highly expressive discriminator tends to learn a non-smooth function overfitting w.r.t. training data points sparsely populated in high-dimensional space. On this aspect, we take inspiration from the literature of adversarial learning of deep neural nets, where high susceptibility of models to small noise, imperceptible to humans, is a common problem. Such a phenomenon is observed despite standard regularization techniques, such as weight decay, dropout, batch normalization, etc. Quoting Yoshida and Miyato (2017), "adversarial training is designed to achieve insensitivity to the perturbation of training data" (Goodfellow et al., 2015; Zhao et al., 2020; Dong et al., 2020). While for the problem of TE estimation, there is neither an "adversary", nor a need to generalize to unseen domains (Volpi et al., 2018), the important takeaway is that the functional outputs of an expressive discriminator must be regularized to be consistent w.r.t. the perturbations to its input, thus ensuring local Lipschitz-like properties of the output function (Yang et al., 2020; Jiang et al., 2020). This allows a safe and robust in-sample estimation of TE.

The challenge is to have an appropriate choice of a perturbation model. A popular choice to perturb data is by adding Gaussian noise. We argue in favor of an even more general perturbation model, which may be agnostic to the data distribution while respecting the underlying data manifold locally (since the desired smoothness of model outputs is *local* in input space), and preferably non-stationary w.r.t. input space. Accordingly, we propose a novel perturbation model, based on *locality sensitive hashing (LSH)*, which is a well known technique for finding nearest neighbors in high dimensions (Indyk and Motwani, 1998; Kulis and Grauman, 2009; Grauman and Fergus, 2013; Zhao et al., 2014; Wang et al., 2017). As per the theory of LSH, a hashcode represents a local neighborhood in the input space, characterizing the underlying data manifold locally. We propose that the outputs of a discriminative model should be consistent within a hashcode bin (one can think of it as a histogram bin in high dimensions), which is accomplished by generating perturbations local to a bin. Sampling perturbations from a hashcode bin doesn't introduce any bias since the hashcode bins correspond to data-driven histograms capable of characterizing the underlying true data distribution (Lugosi and Nobel, 1996) as we show in our theoretical analysis of the estimator (Sec. 3.2). For practical purposes, perturbations are generated from a convex combination of the existing data points from the same bin (i.e. sampling from within a convex hull of the data points); see Fig. 2.

In this paper, we define a simple yet effective information theoretic measure to ensure the consistency of the model outputs, i.e. *minimize conditional entropy* of the model output given the locality sensitive hashcodes of perturbed inputs. The rest of this paper is organized as follows. After discussing the basics of TE and related work in Sec. 2, we present a novel TE estimator in Sec. 3 along with theoretical guaranties (Sec. 3.2). A thorough empirical analysis using a synthetic dataset, a neuroscience dataset of activity in different brain regions, and two financial datasets of high frequency trading activity in US stocks is provided in Sec. 4.

2 BACKGROUND & RELATED WORKS

Transfer entropy, introduced originally by Schreiber (2000), refers to the reduction in uncertainty for forecasting a time series given the knowledge of another time series.

Let X, Y be two discrete- or real-valued time series defined over some probability space $(\Omega, \mathcal{F}, \mathbb{P})$. Let $\mathcal{X}_t$, $\mathcal{Y}_t$ be the random variables denoting X, Y at time t respectively, where $t \in \{0, 1, \ldots, \}$, and let x_t, y_t be their observed values. Furthermore, let $\boldsymbol{\mathcal{Y}}_t$ denote the t-dimensional vector $\boldsymbol{\mathcal{Y}_t} \equiv (\mathcal{Y}_0, \cdots, \mathcal{Y}_{t-1})$; and let $\boldsymbol{y}_t$ be a realization of $\boldsymbol{\mathcal{Y}}_t$. Then, the conditional entropy of $\mathcal{Y}_t$, given its past observations, i.e., the uncertainty in forecasting Y for the current timestep, conditioned on its history, is represented as below.

$$\mathcal{H}\left(\mathcal{Y}_t|\mathcal{Y}_0,\cdots,\mathcal{Y}_{t-1}\right) \equiv \mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) \quad (1)$$

$$\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) = \int_{\boldsymbol{\mathcal{Y}}_t} p(\boldsymbol{y}_t)\log p(y_t|\boldsymbol{y}_{t-1})d\boldsymbol{y}_t \quad (2)$$

Uncertainty when forecasting time series Y, given the knowledge of both its own past as well as X's previous realizations, is expressed as the following conditional entropy:

$$\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1},\boldsymbol{\mathcal{X}}_{t-1}).$$

Mathematically, TE is the difference between two conditional entropy terms, measuring the additional information on $\mathcal{Y}_t$ available in the past realizations of X, that is not already present in the past of Y.

$$\mathcal{T}_{X\to Y} = \mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) - \mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1},\boldsymbol{\mathcal{X}}_{t-1}) \geq 0 \quad (3)$$

2.1 RELATED WORKS: TE ESTIMATORS

A popular approach for estimating TE is based on k nearest neighbors (Lindner et al., 2011; Vicente and Wibral, 2014; Zhu et al., 2015; Wollstadt et al., 2018). , which measures TE indirectly through joint entropy (Kozachenko and Leonenko, 1987; Zhao and Lai, 2020), so Eq. 3 must be re-written as,

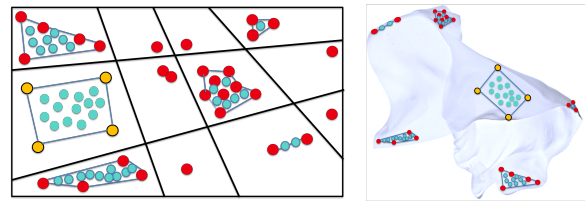
$$\begin{aligned}\mathcal{T}_{X\to Y} &= \mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) - \mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1},\boldsymbol{\mathcal{X}}_{t-1}) \quad (4)\\ &= \mathcal{H}(\boldsymbol{\mathcal{Y}}_t) - \mathcal{H}(\boldsymbol{\mathcal{Y}}_{t-1}) - \mathcal{H}(\boldsymbol{\mathcal{Y}}_t,\boldsymbol{\mathcal{X}}_{t-1}) + \mathcal{H}(\boldsymbol{\mathcal{Y}}_{t-1},\boldsymbol{\mathcal{X}}_{t-1}).\end{aligned}$$

Due to significant different scales of distances across these four terms, the error biases do not cancel each other. Attempts to correct the compounding of biases by estimating joint entropy terms together, using nearest neighbors in the joint space of all the variables (Kraskov et al. (2004), Lindner et al. (2011)) cannot adequately address vulnerability to the dependency based noise.The above formulation is also problematic when Y has a long memory, and consequently, the conditional random variable, $\boldsymbol{\mathcal{Y}}_{t-1}$, is high dimensional, with noisy dependencies w.r.t. the target variable $\mathcal{Y}_t$; for example, when only a few of the many dimensions in $\boldsymbol{\mathcal{Y}}_{t-1}$ explain $\mathcal{Y}_t$, while the other ones are noise components. Same applies for noisy dependencies of $\boldsymbol{\mathcal{X}}_{t-1}$ w.r.t. $\mathcal{Y}_t$, when estimating $\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1},\boldsymbol{\mathcal{X}}_{t-1})$. When estimating TE between time series from real world domains like finance and neuroscience, it is natural to expect such long ranged, noisy temporal dependencies.

Kernel density or histogram-based estimators (Ver Steeg and Galstyan, 2012; Zuo et al., 2013) also suffer from dependency based noise. Even without noise, these estimators are efficient only in low dimensions. In a recent work, Ma (2019) show TE to be equivalent to Copula entropy, but their estimator relies upon estimating joint entropy in high dimensions, which is a hard problem on its own.

TE can also be expressed in terms of conditional mutual information:

$$\begin{aligned}\mathcal{T}_{X\to Y} &= \mathcal{I}(\boldsymbol{\mathcal{X}}_{t-1}:\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1})\\ &= \mathcal{I}(\boldsymbol{\mathcal{X}}_{t-1}:\mathcal{Y}_t,\boldsymbol{\mathcal{Y}}_{t-1}) - \mathcal{I}(\boldsymbol{\mathcal{X}}_{t-1}|\boldsymbol{\mathcal{Y}}_{t-1}). \quad (5)\end{aligned}$$



(a) 2-D space. (b) 2-D manifold.

Figure 2: This figure illustrates the idea of hashcode-based adversarial regularization. In 2(a), data points (*Red* or *Yellow* dots) are dispersed in 2-D space; the lines represent hash functions, and their intersections correspond to *hashcode bins*. Each bin represents a local neighborhood in the input space, data points in a bin being neighbors of each other. Owing to the locality of a hashcode bin w.r.t. the data manifold, perturbations are generated (*Blue* dots) in a bin from randomly sampled convex combinations of the existing data points. We propose that model outputs for the perturbations within a hashcode bin should be consistent w.r.t. each other. In 2(b), hashcode bins from 2(a) are shown on a manifold embedded in 3-D space, illustrating how hashcode bins represent the data manifold locally, thereby leading to a non-stationary perturbation function. For instance, the bin corresponding to yellow dots has a data distribution of highest entropy, thus implying perturbations of the greatest magnitude in the bin.

While theoretically appealing, it is notoriously difficult to estimate mutual information between two high dimensional variables. Zhang et al. (2019) propose to estimate TE through Eq. 5 using the deep neural nets based estimator of mutual information (MINE) due to Belghazi et al. (2018). McAllester and Stratos (2020) show that MINE has high variance which increases with true mutual information value itself, which renders it unsuitable for estimating TE.

Considering these limitations, we introduce a novel estimator of TE in Sec. 3.

3 TRANSFER ENTROPY ESTIMATOR

Referring back to Eq. 3 in Sec. 2, we propose to estimate transfer entropy (TE) via the direct estimation of the *conditional entropy* terms. A direct empirical estimate of the conditional entropy term, $\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1})$, is expressed as,

$$\hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) = -\frac{1}{n}\sum_{i=1}^{n}\log p(y_t^{(i)}|\boldsymbol{y}_{t-1}^{(i)}). \quad (6)$$

Identical estimation logic applies for $\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1},\boldsymbol{\mathcal{X}}_{t-1})$. Here, we make an important observation: a discriminative model that is trained by maximizing the conditional log likelihood of the target variable given the input variable can be employed as an estimator of conditional entropy.

For discrete-valued time series with support set C, a classifier trained by cross entropy loss can provide an empirical estimate of conditional entropy itself.

$$\hat{\mathcal{H}}_q(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) = \frac{1}{n}\sum_{i=1}^{n} -\sum_{c\in C} \mathbb{I}_{y_t^{(i)}=c} \log f(\boldsymbol{y_{t-1}^{(i)}})[c]$$
$$= \frac{1}{n}\sum_{i=1}^{n} -\log q(y_t^{(i)} = c|\boldsymbol{y}_{t-1}^{(i)}), \quad (7)$$

where $\mathbb{I}(.)$ is an indicator function; $f_c(\boldsymbol{y_{t-1}^{(i)}})$ is output of the classifier, a multinomial vector of inferred class probabilities; $\log q(y_t^{(i)}|\boldsymbol{y}_{t-1}^{(i)})$ is an estimate of conditional log likelihood. By Jensen's inequality, it is well known that a proposal distribution $q(.)$ for $p(.)$ upper bounds the corresponding entropy function, $\mathcal{H}_q \geq \mathcal{H}_p$, with the error bias being KL-divergence between the two distributions, $\mathcal{D}_{KL}(p(.|.)||q(.|.)) > 0$ (Cover, 1999).

For continuous valued time series, any regression model trained with mean squared error as the loss function can be employed to estimate conditional entropy if errors are assumed to be Gaussian distributed.

$$y_t \sim \mathcal{N}(f(\boldsymbol{y}_{t-1}), \sigma) \quad (8)$$

Here, $f(\boldsymbol{y}_{t-1})$ can be a highly expressive deep neural net which can essentially approximate any functional relationship between target y_t and input $\boldsymbol{y}_{t-1}$.

Therefore, a discriminator can act as a conditional entropy estimator, with the advantage of being efficient even in very high dimensional noisy feature spaces. This estimator has an *error bias* naturally inherited by the model bias in the discriminator. In theory, the error bias for estimating TE with conditional likelihood $q(.|.)$ stemming from a discriminator, as opposed to true conditional likelihood $p(.|.)$, is as below.

$$\mathcal{T}^q_{X\to Y} - \mathcal{T}_{X\to Y} = \mathcal{D}_{KL}(p(y_t|\boldsymbol{y}_{t-1})||q(y_t|\boldsymbol{y}_{t-1}))$$
$$-\mathcal{D}_{KL}(p(y_t|\boldsymbol{y}_{t-1}, \boldsymbol{x}_{t-1})||q(y_t|\boldsymbol{y}_{t-1}, \boldsymbol{x}_{t-1})) \quad (9)$$

The two terms in the r.h.s. are KL-divergence terms which are error biases of $\mathcal{H}^q(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1})$ and $\mathcal{H}^q(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1})$ respectively. Since both terms are non-negative, $\mathcal{D}(.||.)) \geq 0$, they *counteract* each other leading to a smaller magnitude of the overall error bias of TE. Theoretically, one should expect the error bias of $\mathcal{H}^q(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1})$ to be larger than or equal to its counterpart due to the conditioning upon a higher number of dimensions; this should lead to a net negative error bias. In addition, there is also a bias due to the finite sample size for both of the conditional entropy terms, which could be positive or negative; we analyze variance of the estimates due to finite samples size in Sec. 3.2.

Machine learning literature provides a plethora of discriminative models for time series modeling, such as temporal convolution networks (Bai et al., 2018), N-Beats (Oreshkin et al., 2019), etc, which can be employed in our TE estimator. The challenge, however, is that an expressive discriminator like a deep neural net, with an ability to learn any function, can overfit to training data, even under standard generalization techniques such as weight decay, dropout, etc. In sparsely populated regions of the underlying manifold of model inputs, the output function may be non-smooth.

Inspired by recent works in adversarial training of deep neural nets (Goodfellow et al., 2015; Zhao et al., 2020; Dong et al., 2020), we propose to accomplish Lipschitz-like smoothness of the output function by ensuring that model outputs are consistent w.r.t. perturbations in inputs. A good choice for a perturbation model should be able to characterize the underlying data manifold locally, since the perturbations are supposed to be local w.r.t. inputs.

3.1 GENERATING PERTURBATIONS VIA LSH

The general concept of data perturbations based regularization can be formalized as below.

$$\bar{\boldsymbol{y}} \sim g(\boldsymbol{y}) \quad (10)$$

Here, $g(.)$ is a model that generates perturbations for a given input $\boldsymbol{y}$. An explicit way to ensure that the model output function, $f(.)$, is consistent w.r.t. perturbations in input, i.e., $f(\boldsymbol{y}) = f(\bar{\boldsymbol{y}})$, is to define a regularization penalty for inconsistent model outputs on perturbations. Later in this section, we present an information theoretic regularization criterion which expects model outputs on perturbations of a given input to be of low entropy. One can even employ a non-invasive regularization by tuning the hyper-parameters of a model. An implicit way to ensure consistency is to augment the training data with perturbed data points. Data augmentation is not used here to generalize to unseen domains, but to ensure that the learned model output function is smooth in the local vicinity of training data points, especially if those points were sparsely populated in the input space.

A perturbation model challenges the primary (discriminative) model by perturbing its inputs locally in the data manifold. It can be parametric or non-parametric: perturbations based on Gaussian noise are described in (Rothfuss et al., 2019; Maaten et al., 2013; Bishop, 1995). A good choice of perturbation model should characterize the data manifold locally, whereas the primary model may be inefficient at modeling the local manifold. While not necessary for practical purposes, if the perturbation model can also characterize the underlying true data distribution, $p(\boldsymbol{y}_{t-1}, y_t)$, it would theoretically ensure that there is no error bias from using the perturbations based regularization or data augmentations for estimating TE as we show in Sec. 3.2.

We propose a perturbation model based upon *locality sensitive hashing* (LSH), which perturbs inputs in local neighborhoods of input space. Such neighborhoods are defined from the hashcodes that split the input space into different regions. LSH is a randomized algorithm, that is proven to be efficient in finding nearest neighbors in very high dimensions (Indyk and Motwani, 1998; Zhao et al., 2014; Wang et al., 2017). The core idea is that similar data points according to some distance metric are assigned the same hashcodes with probability inversely proportional to the distance metric.

This theoretical property of hashcodes implies that a hashcode bin represents a local neighborhood (manifold) in input space. We propose a hashcode-based regularization such that model outputs are smooth w.r.t. perturbations of inputs within a hashcode bin. Since a hashcode bin represents the data manifold *locally*, we generate perturbations from randomly sampled *convex combinations* of the existing data points in a bin (See Fig. 2 for an illustration). One advantage of this approach is that the perturbation model is non-stationary w.r.t. the input space, since the perturbations are generated locally w.r.t. hashcode bins. Furthermore, the perturbation model does not make any parametric assumption about the global distribution of data or the data manifold. Mathematically, Eq. 10 can be re-expressed for the hashcodes based perturbation function as follows.

$$\bar{\boldsymbol{y}} \sim g(\boldsymbol{c}) \; s.t. \; \boldsymbol{h}(\boldsymbol{y}) = \boldsymbol{h}(\bar{\boldsymbol{y}}) = \boldsymbol{c}, \tag{11}$$

where, $\boldsymbol{h}(.)$ is a locality sensitive hash function, represented by a set of H hash functions, each outputting one bit, mapping an input $\boldsymbol{y}$ to its hashcode $\boldsymbol{c} \in \{0,1\}^H$. The perturbation model $g(.)$ samples a perturbation w.r.t. a hashcode bin, and not a single input. In practice, it is efficient to sample all the perturbations together across all the hashcode bins.

A pseudo code is presented in Alg. 1. The input data points $(\boldsymbol{y}^{(1)}, \cdots, \boldsymbol{y}^{(n)})$ are the inputs of a model that we regularize, i.e. a discriminator for our problem. First, we compute hashcodes for all the data points and sort the data into their corresponding unique hashcode bins. For each bin, a Dirichlet distribution is then initialized with hyperparameter α, and of dimension equal to the number of data points in the bin. For n_i data points in the i_{th} bin, we sample $n_i b$ number of perturbations from that bin. Each perturbation is sampled in-turn by randomly drawing a multinomial vector from the Dirichlet distribution, which acts as a random convex combination of all the points in the bin.

Since we are interested in local smoothness given by hashcodes bins representing local neighborhoods, we propose a regularization criterion based on information theory. In particular, we minimize the conditional entropy of the model outputs given the hashcode representation of the inputs:

$$\min_{f(.)} \mathcal{H}(f(\boldsymbol{\mathcal{Y}})|\boldsymbol{h}(\boldsymbol{\mathcal{Y}})). \tag{12}$$

Although $f(.)$ & $\boldsymbol{h}(.)$ are deterministic, both $f(\boldsymbol{\mathcal{Y}})$ & $\boldsymbol{h}(\boldsymbol{\mathcal{Y}})$ are stochastic given their dependence on $\boldsymbol{\mathcal{Y}}$. Empirical estimate of this regularization term is easy and cheap to compute. For each hashcode bin, we compute the model outputs for existing as well as the sampled perturbations from Alg. 1, and then we compute an empirical estimate of entropy of those outputs within the bin. Since the target variable is one-dimensional in our problem, i.e. observation y_t for timestep t, computing entropy of the model outputs is easy even for the case of conditionals densities; one can, for example, use non-parametric estimators like histograms. This way, we iterate through all the bins to finally compute the conditional entropy term, $\mathcal{H}(f(\boldsymbol{\mathcal{Y}})|\boldsymbol{h}(\boldsymbol{\mathcal{Y}}))$.

Algorithm 1 Generate Perturbations via LSH

Require: $\{\boldsymbol{y}^{(1)}, \cdots, \boldsymbol{y}^{(n)}\}$, α, b
$\boldsymbol{c}^{(1)}, \cdots, \boldsymbol{c}^{(n)} \leftarrow$ computeHashcode($\boldsymbol{y}^{(1)}, \cdots, \boldsymbol{y}^{(n)}$)
$\boldsymbol{Y}^{(1)}, \cdots, \boldsymbol{Y}^{(m)} \leftarrow$ hashcodeBin($\{(\boldsymbol{y}^{(1)}, \boldsymbol{c}^{(1)})\}_{i=1}^{n}$) *% $\boldsymbol{Y}^{(i)}$ has all the inputs from i_{st} hashcode bin*
for $i = 0 \rightarrow m$ **do**
  $n_i \leftarrow$ countSamplesInBin($\boldsymbol{Y}^{(i)}$)
  $\bar{n}_i \leftarrow n_i * b$ *% no. of perturbations in i_{th} bin*
  for $j = 0 \rightarrow \bar{n}_i$ **do**
    $\boldsymbol{\pi}_j \sim Dir(\alpha \mathbf{1}_{n_i})$ *% sample weights as a convex combination, from Dirichlet distribution*
    $\bar{\boldsymbol{y}}_j^{(i)} \leftarrow \boldsymbol{\pi}_j * \boldsymbol{Y}^{(i)}$ *% generate perturbation in the bin*
  end for
end for
Return $\{(\bar{\boldsymbol{y}}_j^{(i)}, \boldsymbol{c}_j^{(i)})\}_{i=1,j=1}^{m,n_i}$

Another advantage of this approach is that it admits *any* LSH algorithm. We also propose a novel, greedy algorithm for unsupervised learning of locality sensitive hash functions and use for our experiments. See Appendix for details.

3.2 THEORETICAL ANALYSIS

As discussed above, we use locality sensitive hashing (LSH) based regularization for learning the conditional log likelihood estimates by perturbing the inputs within the same hashcode bin. Perturbation may lead to a different distribution than the data distribution, and thus the conditional likelihood estimates derived from this distribution may be biased. We establish the conditions under which the perturbed distribution yields consistent estimates. Let $g_{n,H}(.)$ denote the histogram distribution obtained by using H locality sensitive hash functions and n samples. We will see LSH based data generation as sampling from a data-driven histogram. Using this insight, results from Lugosi and Nobel (1996) and a proof technique similar to Rothfuss et al. (2019), we demonstrate consistency of our sampling approach under some regularity conditions.

Theorem 1. *Let* $\lim_{n\to\infty} \frac{2^H}{n} \rightarrow 0$, $\lim_{n\to\infty} \frac{tH \log n}{n} \rightarrow 0$ *and the input space,* $\boldsymbol{y} \in \mathbb{R}^t$ *is bounded. Consider any function,* $f : \boldsymbol{y} \rightarrow (0, \infty)$ *with* $\log f$ *having finite second order moment w.r.t to* p *and* $g_{n,H}$*. Then,*

$$\lim_{n\to\infty} |\mathbb{E}_p[-\log f(\boldsymbol{y})] - \mathbb{E}_g[-\log f(\boldsymbol{y})]| \rightarrow 0. \tag{13}$$

The above result establishes that the perturbed distribution will yield same estimates in expectation as n becomes large. We provide a formal proof for this in Appendix A.1. As a corollary, we can establish the consistency of our TE estimator. We can rewrite,

$$\mathcal{T}^{q}_{X\rightarrow Y} = \mathbb{E}_p \log \frac{q(y_t|\boldsymbol{y_{t-1}}, \boldsymbol{x_{t-1}})}{q(y_t|\boldsymbol{y_{t-1}})}, \tag{14}$$

and the TE estimated using $g_{n,H}$ be,

$$\mathcal{T}^{q,g}_{X\rightarrow Y} = \mathbb{E}_g \log \frac{q(y_t|\boldsymbol{y_{t-1}}, \boldsymbol{x_{t-1}})}{q(y_t|\boldsymbol{y_{t-1}})}. \tag{15}$$

Corollary 1. *If the conditions in Thm. 1 hold, and the model distribution, $q > 0$ everywhere. Then letting $f(\boldsymbol{y_t}, \boldsymbol{x_{t-1}}) = \frac{q(y_t|\boldsymbol{y_{t-1}}, \boldsymbol{x_{t-1}})}{q(y_t|\boldsymbol{y_{t-1}})}$, we have,*

$$\lim_{n\to\infty} |\mathcal{T}^{q}_{X\to Y} - \mathcal{T}^{q,g}_{X\to Y}| \to 0. \tag{16}$$

Remark: To find a good discriminator q, we optimize the LSH regularized MLE objectives, i.e., $\min_q -\mathbb{E}_{g_{n,H}} \log q(y_t|.)$. As n becomes large, this is the same as computing expectation over data distribution p due to Thm 1, $\min_q -\mathbb{E}_p \log q(y_t|.)$. If the function class is expressive enough, such as a neural network for which the universal approximation theorem holds, the optimal discriminator would correspond to the correct conditional distribution derived from population distribution (Lu and Lu, 2020).

The above results characterize the distribution of perturbed samples and the behaviour of estimates under that distribution. Another aspect of our approach is that it relies on finite sample size, and thus, we next characterize sample complexity of our estimator to obtain high-confidence estimates.

Theorem 2. *For some data distribution p and conditional model distribution q and $-\log q(\boldsymbol{y}_t|(.)) \in [-Q, Q]$. Let $\hat{\mathcal{T}}^{q}_{X\to Y}$ denote the n-sample estimate of transfer entropy. Then with probability $1-\delta$ ($\delta > 0$), we have*

$$|\hat{\mathcal{T}}^{q}_{X\to Y} - \mathcal{T}^{q}_{X\to Y}| \leq 2Q\sqrt{(2/n)\ln(4/\delta)} \tag{17}$$

As a consequence of the above result, we can bound the error variance as below:

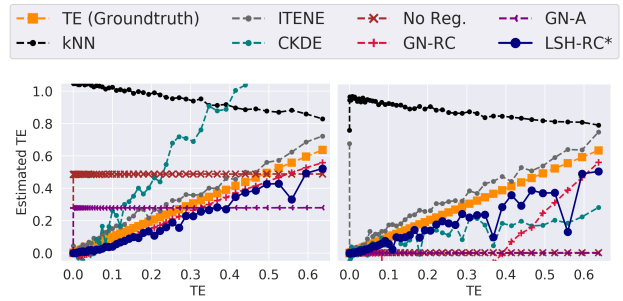
$$\mathbb{E}(\hat{\mathcal{T}}^{q}_{X\to Y} - \mathcal{T}^{q}_{X\to Y})^2 \leq 4Q^2((1-\delta)\frac{2}{n}\ln\frac{4}{\delta} + \delta) \tag{18}$$

The first term is the dominant term in above expression and thus, the variance of the estimator reduces at the rate of $\frac{1}{n}$. See Sec. A.1 for the proofs.

4 EMPIRICAL EVALUATION

In this section, we present empirical results to illustrate the performance of our algorithm in estimating transfer entropy (TE). To demonstrate the efficacy of the method when the ground truth is known, we evaluate TE on a synthetic dataset (Sec. 4.1). We also measure it extensively on two real world examples: a neuroscience dataset (Sec. 4.2) and a dataset representing US stock market activity (Sec. 4.3).

Estimators for Comparison We evaluate four different baseline estimators of TE from the literature: (i) kNN estimator: *kNN*; (ii) Conditional kernel density estimation: *CKDE*; (iii) Copula entropy based estimator of Ma (2019), referred as *Copent*. (iv) Conditional mutual information based estimator of Zhang et al. (2019), referred as *ITENE*. To estimate TE in terms of conditional entropies directly using a discriminative model, we employ two neural models: Temporal Convolution Networks (TCN) and *N-Beats*, as well as Gradient Boosted Machines (GBM). LSH indicates that it is a perturbation model based on locality sensitive hashing (LSH):



(a) Synthetic data ($t = 10$) (b) Synthetic data ($t = 20$)

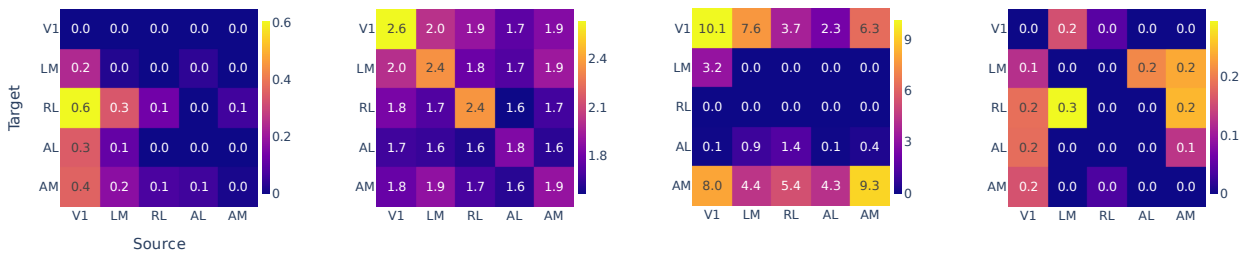
Figure 3: Estimates of TE are plotted w.r.t. the groundtruth for a synthetic dataset; t refers to the time lag for how far back we look into the past to forecast for the current timestep. Here, the suffix "*" refers to the proposed TE estimator.

LSH-RC imposes regularization penalty for inconsistent model outputs, while LSH-A involves data augmentation. The GBM model uses LSH-RC for tuning a large set of hyper-parameters (GBM-LSH-RC*). For neural models, we use LSH-A (TCN-LSH-A* & NBeats-LSH-A*). For the baseline of Gaussian noise as a perturbation model, we likewise have GN-RC and GN-A. If no perturbation model is used, it is referred to as "No Reg.", another baseline. Standard regularization techniques like weight decay and dropout are used for all models, including "No Reg.". For continuous time series, the models are used as regressors trained with mean squared loss, and for discrete-valued time series as classifier with cross entropy loss. Each model has its own strength depending on the nature of data, so in some cases, we present results for the best of the three discriminative models (GBM, TCN, NBeats) accordingly.

Parameter Settings In the perturbation model, we generate new samples 10 times the original number of samples, i.e. b=10 in Alg. 1 (for LSH-A, $b = 3$). For sampling from Dirichlet distribution in Alg. 1, α=0.1. Number of hash functions for LSH is 15, $H = 15$. These parameters do not require fine-tuning, so they were set manually.

4.1 EVALUATION ON SYNTHETIC DATA

Our algorithm for generating binary valued synthetic data is as follows. First we draw 3,000 samples for the target variable, y_t, s.t. $p(y_t = 1) = 0.5$. We assume zero temporal dependencies within time series Y, i.e. $H(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) = H(\mathcal{Y}_t) = \log(2)$. Next, we randomly select one of the t dimensions of the conditioned variable $\boldsymbol{x}_{t-1}$, denoted by x_r, such that it depends on the target y_t, and the rest of the $t-1$ dimensions are independent of y_t; $p(x_r = 1|y_t = 1) = q, p(x_r = 1|y_t = 0) = 1-q$. This implies, $H(\mathcal{Y}_t|\boldsymbol{\mathcal{X}}_{t-1}, \boldsymbol{\mathcal{Y}}_{t-1}) = -q\log q - (1-q)\log(1-q)$, which is basically the entropy of a biased coin with probability of head equal to q. Overall, $\mathcal{T}_{X\to Y} = \log(2) - q\log q - (1-q)\log(1-q)$. For $q = 0$ or $q = 1$, $\mathcal{T}_{X\to Y}$ attains its maximum value of $\log(2)$. For $q = 0.5$, that is using an unbiased coin to sample x_r given y_t, there is no transfer of entropy,



(a) NBeats-LSH-A* (b) kNN (c) Copent (d) ITENE

Figure 4: Estimates of transfer entropy between mouse visual areas using different methods. The matrices shows the transfer entropy values $\mathcal{T}_{\text{Source}\to\text{Target}}$ with source regions along columns and target regions along rows. The brain regions are sorted by ascending hierarchical order from left to right and top to bottom. V1 can be seen as the gateway of the visual system. The result of our method in (a) reveals the brain structure that matches current knowledge of the hierarchical visual system, while others do not.

$\mathcal{T}_{X\to Y} = 0$. We generate synthetic data for varying values of q, from $q = 0$ to $q = 0.5$. The most important aspect of this data generation step is that only *one* of the many dimensions in $\boldsymbol{x}_{t-1}$ is dependent on y_t, while the others are not. In Sec. 2, we referred to this as *noisy dependency*.

In Fig. 3, we present experimental results for the estimation of TE in the synthetic dataset.[1] Here, we only use GBM model as a discriminator. The figures differ by the dimension of the conditioned variables ($t = 20$ and $t = 10$), $\boldsymbol{x}_{t-1}$ or $\boldsymbol{y}_{t-1}$, i.e. how far back we look into the past for forecasting time series Y. In both cases, the kNN-based estimator provides a severe overestimate of TE, and is almost agnostic to the dependency of x_r on y_t, driven significantly by noisy signal from all the other $2t-1$ dimensions. Its error reduces when dependency between x_r and y_t approaches the highest value. Despite the popularity of the estimator, the results are unsurprising for the aforementioned technical reasons. CKDE estimates correlate to the ground truth values of TE, but with a significant error bias. ITENE obtains the TE estimates with very low error bias.[2] Our approach of LSH-RC* also has very low estimation errors, although there are a few instances where it is high. One challenge is to optimize the trade-off between the log likelihood objective and the regularization term. In contrast, the baseline approaches of directly using a discriminator without regularization (No Reg.), and the Gaussian noise based data augmentation are both completely unsuccessful. Regularization using Gaussian noise based data perturbation model (GN-RC) seems to work for smaller time lag of $t = 10$.

It is worth noting that the above described process of generating synthetic data is not artificial, but recreates highly noisy temporal dependencies observed between and within time series from domains such as neuroscience and finance.

[1]Copent estimator is excluded from the figure since its estimates are way beyond the range of true TE.

[2]We observed that variance of the TE estimates from ITENE increases if we were to consider longer time lag (say, $t = 100$).

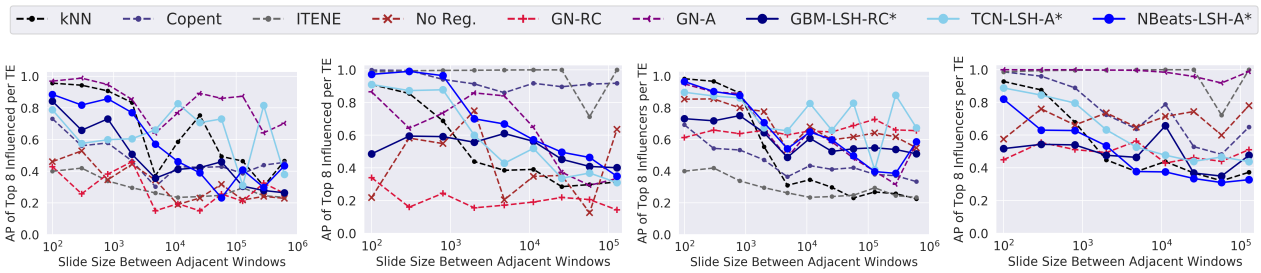
4.2 EVALUATION ON NEUROSCIENCE DATA

Next, we applied the method to the neuroscience dataset, Allen Brain Observatory–Visual Coding Neuropixels (Siegle et al., 2021), by offering a metric from information theory perspective to discover the structure of the mouse brain and verify whether the results agree with the current findings of the visual system. Both anatomical and functional studies have shown that the brain visual system is hierarchically organized and the visual information propagates across the cortical areas in order accordingly (Siegle et al., 2021; Harris et al., 2019). During the early visual process, it is expected to see that the activities of low-order regions drive those of high-order regions.

Fig. 4 presents the results, showing estimated TEs between regions. Details of data processing are in Appendix C. The columns indicate the source regions and the rows indicate the target regions. The hierarchical order of the brain regions, from low to high, is V1, LM, RL, AL, and AM (Harris et al., 2019; Siegle et al., 2021), which are sorted along rows and columns. A larger value indicates the source region contributes more significantly to the target region's entropy, which implies the direction of information flow and the source region has an impact on the activity of the downstream target region. In Fig. 4(a), all large values concentrate in the bottom left triangle, which means the low-order regions impact the high-order regions, thus the conclusion agrees with the hierarchical order found by other anatomical or functional methods. In contrast, other methods in Fig. 4(b), (c), and (d) do not properly reveal the hierarchical relationships among the visual areas, especially they present many large positive TE values in the top right triangle matrix. For example, these methods show $\mathcal{T}_{\text{AM}\to\text{V1}} > 0$, meaning AM impacts V1, which is not bio-physically reasonable. More analysis can be found in Appendix C.

4.3 EVALUATION ON US STOCK DATA

We consider the top 64 most actively traded stocks in the US, and define two sets of time series for TE estimation. The first concerns the frequency of order arrivals during regular intervals: for a fixed interval of 100 milliseconds, we count the number of all orders for each stock over the course of 6



(a) Liquidity: Influenced securities (b) Mid-Price: Influenced securities (c) Liquidity: Influencer securities (d) Mid-Price: Influencer securities

Figure 5: US Stocks: Average precision is computed as a measure of consistency between transfer entropy estimates from two adjacent time windows. Here, suffix "*" refers to the proposed TE estimators.

trading days. The second set of time series constitutes the observed mid-price (MP) changes over 1 second intervals for each of the same 64 stock, over a 14 day-period. In both settings, we consider context size of 120 timesteps to forecast the current timestep ($t = 120$).

We use a window of 3000 timesteps (5 minutes) for estimating transfer entropy between each pair of 64 stocks; for the case of mid-price temporal dynamics, a window is of one hour (3600 timesteps). We construct many such windows of identical size during the period of 6 days (or 14 days), so that the time gap between two subsequent windows increases exponentially in time. We expect transfer entropy estimations from two windows close in time to be similar/ consistent to each other, and significantly different, or inconsistent, as they are further apart in time (due to the inherent heteroskedasticity of competitive markets).

In this context, transfer entropy can be represented as a sequence of 64-by-64 matrices. We use two criteria for such an evaluation of consistency versus inconsistency. For a given security, we evaluate if the top 8 securities influenced by it are consistent between two adjacent windows (row-wise consistency). Denoting transfer entropy matrices from two adjacent windows as $\boldsymbol{T}^{(1)}$ and $\boldsymbol{T}^{(2)}$, we set the top 8 values in each row of $\boldsymbol{T}^{(1)}$ as a positive label and the rest as negative labels, and then we use $\boldsymbol{T}^{(2)}$ as scores w.r.t. the labels from $\boldsymbol{T}^{(1)}$, so as to compute Average Precision (AP) of the top 8 influenced securities. We expect this score to drop as we increase the time gap between two adjacent windows, as the list of top securities influenced by a security should change as market conditions evolve.In Fig. 5(a) and 5(b), we present the results for this evaluation criterion. For a slide size of up to 1000 timesteps between two adjacent windows, we expect a very high AP score (high consistency), since it is reasonably small compared to the window size of 3000 timesteps (or 3600 timesteps). As we increase the size to be a large multiple of the window size, average precision should drop, and then it may remain small or decrease further. NBeats-LSH-A* outperforms all other estimators,[3], following the expected pattern of consistency for the criterion explained above. Estimations with no regularization or Gaussian noise based perturbation (GN-RC & GN-A) lead to low consistency regardless of the slide size; in Fig. 5(a), GN-A seems to provide high consistency for all the slide sizes which is not a desired pattern of consistency either. The consistency scores for the kNN estimator remain high for small slide sizes, but drops sharply, although sometimes it provides high consistency even at large slide sizes. Copent and ITENE estimators have consistency scores almost constant w.r.t. slide size, indicating their vulnerability to noise. The choice of the underlying discriminator in our estimation approach depends upon the phenomenon of interest; for instance, GBM-LSH-RC* performs well for modeling the order activity (discrete valued time series) whereas TCN-LSH-A* is better suited for modeling the temporal dynamics of mid-prices of stocks (continuous valued time series).

Another criterion for evaluation is to see if the top 8 securities influencing a given security are consistent between two adjacent windows. Similar to the previous exercise, we compute average precision but column-wise instead. Results shown in Fig. 5(c) and Fig. 5(d), exhibit similar patterns of superiority of our proposed estimator w.r.t. the baselines.

Overall, the experimental results suggest that transfer entropy estimation can be unreliable when dealing with long ranged and noisy temporal dependencies, as observed in real world domains like finance and neuroscience. Our proposed estimator, regularized using an LSH based perturbation model, shows robustness in the selected experiments.

5 CONCLUSIONS

We established that empirical estimation of transfer entropy between time series is a challenging problem, especially if temporal dependencies are long ranged and noisy. Such noise is common in domains such as finance and neuroscience, though difficult to characterize. We explained theoretical reasons for why well known estimators are prone to such noise, and propose a novel method - a discriminator regularized using a perturbation model based on locality sensitive hashing. We proved consistency of the estimator, and that it's variance decreases linearly in sample size. It is also shown to be efficient empirically in synthetic as well as real world settings of neuroscience and finance domain.

[3]We exclude CKDE from this evaluation since it doesn't scale to high dimensions, and performs poorly on the synthetic datasets.

Bibliography

Bai, S., Kolter, J. Z., and Koltun, V. (2018). An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*.

Barnett, L., Barrett, A. B., and Seth, A. K. (2009). Granger causality and transfer entropy are equivalent for gaussian variables. *Physical review letters*, 103(23):238701.

Barnett, L. and Bossomaier, T. (2012). Transfer entropy as a log-likelihood ratio. *Physical review letters*, 109(13):138105.

Belghazi, M. I., Baratin, A., Rajeshwar, S., Ozair, S., Bengio, Y., Courville, A., and Hjelm, D. (2018). Mutual information neural estimation. In *International Conference on Machine Learning*.

Bishop, C. M. (1995). Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116.

Cicalese, F., Laber, E., and Murtinho, L. (2019). New results on information theoretic clustering. In *International Conference on Machine Learning*, pages 1242–1251. PMLR.

Cover, T. M. (1965). Geometrical and statistical properties of systems of linear inequalities with applications in pattern recognition. *IEEE transactions on electronic computers*, (3):326–334.

Cover, T. M. (1999). *Elements of information theory*. John Wiley & Sons.

Dabney, W., Ostrovski, G., Silver, D., and Munos, R. (2018). Implicit quantile networks for distributional reinforcement learning. In *International conference on machine learning*, pages 1096–1105. PMLR.

Dimpfl, T. and Peter, F. J. (2013). Using transfer entropy to measure information flows between financial markets. *Studies in Nonlinear Dynamics and Econometrics*, 17(1):85–102.

Dong, Y., Deng, Z., Pang, T., Zhu, J., and Su, H. (2020). Adversarial distributional training for robust deep learning. *Advances in Neural Information Processing Systems*.

Garg, S., Galstyan, A., Ver Steeg, G., and Cecchi, G. A. (2019a). Nearly-unsupervised hashcode representations for biomedical relation extraction. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Garg, S., Galstyan, A., Ver Steeg, G., Rish, I., Cecchi, G., and Gao, S. (2019b). Kernelized hashcode representations for relation extraction. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems*.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2015). Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*.

Grauman, K. and Fergus, R. (2013). Learning binary hash codes for large-scale image search. *Machine learning for computer vision*, 411(49-87):1.

Harris, J. A., Mihalas, S., Hirokawa, K. E., Whitesell, J. D., Choi, H., Bernard, A., Bohn, P., Caldejon, S., Casal, L., Cho, A., et al. (2019). Hierarchical organization of cortical and thalamic connectivity. *Nature*, 575(7781):195–202.

He, J. and Shang, P. (2017). Comparison of transfer entropy methods for financial time series. *Physica A: Statistical Mechanics and its Applications*.

Indyk, P. and Motwani, R. (1998). Approximate nearest neighbors: towards removing the curse of dimensionality. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*.

Institute, A. (2020 (accessed 2020)). *Visual Coding - Neuropixels*.

Jiang, Z., Chen, T., Chen, T., and Wang, Z. (2020). Robust pre-training by adversarial contrastive learning. In *NeurIPS*.

Jizba, P., Kleinert, H., and Shefaat, M. (2012). Rényi's information transfer between financial time series. *Physica A: Statistical Mechanics and its Applications*, 391(10):2971–2989.

Joly, A. and Buisson, O. (2011). Random maximum margin hashing. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 873–880. IEEE.

Kass, R. E., Amari, S.-I., Arai, K., Brown, E. N., Diekman, C. O., Diesmann, M., Doiron, B., Eden, U. T., Fairhall, A. L., Fiddyment, G. M., et al. (2018). Computational neuroscience: Mathematical and statistical perspectives. *Annual review of statistics and its application*, 5:183–214.

Kim, S., Ku, S., Chang, W., and Song, J. W. (2020). Predicting the direction of us stock prices using effective transfer entropy and machine learning techniques. *IEEE Access*, 8:111660–111682.

Kingma, D. P. and Welling, M. (2019). An introduction to variational autoencoders. *arXiv preprint arXiv:1906.02691*.

Kozachenko, L. F. and Leonenko, N. N. (1987). A statistical estimate for the entropy of a random vector. *Problems of Information Transmission*.

Kraskov, A., Stögbauer, H., and Grassberger, P. (2004). Estimating mutual information. *Physical Review E*, 69:066138.

Kulis, B. and Grauman, K. (2009). Kernelized locality-sensitive hashing for scalable image search. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2130–2137. IEEE.

Kulis, B. and Grauman, K. (2012). Kernelized locality-sensitive hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1092–1104.

Lindner, M., Vicente, R., Priesemann, V., and Wibral, M. (2011). Trentool: A matlab open source toolbox to analyse information flow in time series data with transfer entropy. *BMC neuroscience*.

Lizier, J. T. (2014). Jidt: An information-theoretic toolkit for studying the dynamics of complex systems. *Frontiers in Robotics and AI*, 1:11.

Lombardi, D. and Pant, S. (2016). Nonparametric k-nearest-neighbor entropy estimator. *Physical Review E*, 93(1):013310.

Lu, Y. and Lu, J. (2020). A universal approximation theorem of deep neural networks for expressing probability distributions. *Advances in neural information processing systems*, 33:3094–3105.

Lugosi, G. and Nobel, A. (1996). Consistency of data-driven histogram methods for density estimation and classification. *The Annals of Statistics*, 24(2):687–706.

Ma, J. (2019). Estimating transfer entropy via copula entropy. *arXiv preprint arXiv:1910.04375*.

Maaten, L., Chen, M., Tyree, S., and Weinberger, K. (2013). Learning with marginalized corrupted features. In *International Conference on Machine Learning*, pages 410–418. PMLR.

McAllester, D. and Stratos, K. (2020). Formal limitations on the measurement of mutual information. In *International Conference on Artificial Intelligence and Statistics*, pages 875–884. PMLR.

Montalto, A., Faes, L., and Marinazzo, D. (2014). Mute: A new matlab toolbox for estimating the multivariate transfer entropy in physiological variability series. In *2014 8th Conference of the European Study Group on Cardiovascular Oscillations (ESGCO)*, pages 59–60. IEEE.

Oreshkin, B. N., Carpov, D., Chapados, N., and Bengio, Y. (2019). N-beats: Neural basis expansion analysis for interpretable time series forecasting. In *International Conference on Learning Representations*.

Rasul, K., Seward, C., Schuster, I., and Vollgraf, R. (2021). Autoregressive denoising diffusion models for multivariate probabilistic time series forecasting. In *International Conference on Machine Learning*, pages 8857–8868. PMLR.

Restrepo, J. F., Mateos, D. M., and Schlotthauer, G. (2020). Transfer entropy rate through lempel-ziv complexity. *Physical Review E*, 101(5):052117.

Rothfuss, J., Ferreira, F., Walther, S., and Ulrich, M. (2019). Conditional density estimation with neural networks: Best practices and benchmarks. *arXiv preprint arXiv:1903.00954*.

Sandoval, L. (2014). Structure of a global network of financial companies based on transfer entropy. *Entropy*, 16(8):4443–4482.

Schreiber, T. (2000). Measuring information transfer. *Physical review letters*, 85(2):461.

Siegle, J. H., Jia, X., Durand, S., Gale, S., Bennett, C., Graddis, N., Heller, G., Ramirez, T. K., Choi, H., Luviano, J. A., et al. (2021). Survey of spiking in the mouse visual system reveals functional hierarchy. *Nature*, 592(7852):86–92.

Singh, S. and Póczos, B. (2016). Analysis of k-nearest neighbor distances with application to entropy estimation. *arXiv preprint arXiv:1603.08578*.

Sipahi, R. and Porfiri, M. (2020). Improving on transfer entropy-based network reconstruction using time-delays: Approach and validation. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 30(2):023125.

Song, J., Kalluri, P., Grover, A., Zhao, S., and Ermon, S. (2019). Learning controllable fair representations. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2164–2173. PMLR.

Song, Y. and Kingma, D. P. (2021). How to train your energy-based models. *arXiv preprint arXiv:2101.03288*.

Steinmetz, N. A., Aydin, C., Lebedeva, A., Okun, M., Pachitariu, M., Bauza, M., Beau, M., Bhagat, J., Böhm, C., Broux, M., et al. (2021). Neuropixels 2.0: A miniaturized high-density probe for stable, long-term brain recordings. *Science*, 372(6539):eabf4588.

Steinmetz, N. A., Koch, C., Harris, K. D., and Carandini, M. (2018). Challenges and opportunities for large-scale electrophysiology with neuropixels probes. *Current opinion in neurobiology*, 50:92–100.

Ursino, M., Ricci, G., and Magosso, E. (2020). Transfer entropy as a measure of brain connectivity: a critical analysis with the help of neural mass models. *Frontiers in computational neuroscience*, 14:45.

Ver Steeg, G. and Galstyan, A. (2012). Information transfer in social media. In *Proceedings of the 21st international conference on World Wide Web*, pages 509–518.

Ver Steeg, G. and Galstyan, A. (2012). Information transfer in social media. In *WWW'12*.

Vergara, J. R. and Estévez, P. A. (2014). A review of feature selection methods based on mutual information. *Neural computing and applications*, 24(1):175–186.

Vicente, R. and Wibral, M. (2014). Efficient estimation of information transfer. In *Directed Information Measures in Neuroscience*, pages 37–58. Springer.

Vicente, R., Wibral, M., Lindner, M., and Pipa, G. (2011). Transfer entropy—a model-free measure of effective connectivity for the neurosciences. *Journal of computational neuroscience*.

Volpi, R., Namkoong, H., Sener, O., Duchi, J., Murino, V., and Savarese, S. (2018). Generalizing to unseen domains via adversarial data augmentation. *Advances in Neural Information Processing Systems*.

Wang, J., Zhang, T., Sebe, N., Shen, H. T., et al. (2017). A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

Wollstadt, P., Lizier, J. T., Vicente, R., Finn, C., Martinez-Zarzuela, M., Mediano, P., Novelli, L., and Wibral, M. (2018). Idtxl: The information dynamics toolkit xl: a python package for the efficient analysis of multivariate information dynamics in networks. *arXiv preprint arXiv:1807.10459*.

Yang, Y.-Y., Rashtchian, C., Zhang, H., Salakhutdinov, R. R., and Chaudhuri, K. (2020). A closer look at accuracy vs. robustness. In *NeurIPS*.

Yoshida, Y. and Miyato, T. (2017). Spectral norm regularization for improving the generalizability of deep learning. *arXiv preprint arXiv:1705.10941*.

Zhang, J., Simeone, O., Cvetkovic, Z., Abela, E., and Richardson, M. (2019). Itene: Intrinsic transfer entropy neural estimator. *arXiv preprint arXiv:1912.07277*.

Zhao, K., Lu, H., and Mei, J. (2014). Locality preserving hashing. In *AAAI*, pages 2874–2881.

Zhao, L., Liu, T., Peng, X., and Metaxas, D. (2020). Maximum-entropy adversarial data augmentation for improved generalization and robustness. *Advances in Neural Information Processing Systems*.

Zhao, P. and Lai, L. (2020). Analysis of knn density estimation. *arXiv preprint arXiv:2010.00438*.

Zhu, J., Bellanger, J.-J., Shu, H., and Le Bouquin Jeannès, R. (2015). Contribution to transfer entropy estimation via the k-nearest-neighbors approach. *Entropy*, 17(6):4173–4201.

Zhu, L. and Laptev, N. (2017). Deep and confident prediction for time series at uber. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pages 103–110. IEEE.

Zuo, K., Zhu, J., Bellanger, J.-J., and Jeannès, R. L. B. (2013). Adaptive kernels and transfer entropy for neural connectivity analysis in eeg signals. *IRBM*, 34(4-5):330–336.

A THEORETICAL PROOFS

A.1 CONSISTENCY OF REGULARIZATION WITH HASH CODES

Below, we present a proof of Thm 1. We can see LSH based data generation as sampling from a data-driven histogram. Using this insight and using results from Lugosi and Nobel (1996) and a proof technique similar to Rothfuss et al. (2019), we will demonstrate consistency of our sampling approach. We use this approach to estimate entropy, i.e., $\mathcal{H}(\boldsymbol{\mathcal{Y}}_t|\boldsymbol{\mathcal{Y}_{t-1}})$.

Theorem 3. ***(Theorem 1 restated)*** *Suppose* $\lim_{n\to\infty} \frac{2^H}{n} \to 0$, $\lim_{n\to\infty} \frac{tH\log n}{n} \to 0$ *and the input space, i.e.,* $\boldsymbol{y} \in \mathbb{R}^t$ *is bounded. Consider any function,* $f : \boldsymbol{y} \to (0, \infty)$ *with* $\log f$ *having finite second order moment w.r.t to* p *and* $g_{n,H}$*, then*

$$\lim_{n\to\infty} |\mathbb{E}_p[-\log f(\boldsymbol{y})] - \mathbb{E}_g[-\log f(\boldsymbol{y})]| \to 0$$

Proof. Let $l(\boldsymbol{\theta}) = -\mathbb{E}_{\boldsymbol{y}\sim p} \log f(\boldsymbol{y})$ and $l_n^{(g,H)}(\boldsymbol{\theta}) = -\mathbb{E}_{\boldsymbol{y}\sim g_{n,H}} \log f(\boldsymbol{y})$. Here $g_{n,H}(\boldsymbol{x})$ denotes the perturbed distribution obtained by using H dimensional hashcode function and n samples. Consider,

$$\begin{aligned}
|l(\boldsymbol{\theta}) - l_n^{(g,d)}(\boldsymbol{\theta})|^2 &= \left|-\mathbb{E}_{\boldsymbol{y}\sim p} \log f(\boldsymbol{y}) + \mathbb{E}_{\boldsymbol{y}\sim g_{n,H}} \log f(\boldsymbol{y})\right|^2 \\
&= \left(\mathbb{E}_{\boldsymbol{y}\sim p} \log f(\boldsymbol{y}) - \mathbb{E}_{\boldsymbol{y}\sim g_{n,H}} \log f(\boldsymbol{y})\right)^2 \\
&= \left(\int d\boldsymbol{y} \log f(\boldsymbol{y})(p(\boldsymbol{y}) - g_{n,d}(\boldsymbol{y}))\right)^2 \\
&= \left(\int d\boldsymbol{y} \left(\log f(\boldsymbol{y})(p(\boldsymbol{y}) - g_{n,d}(x))^{0.5}\right) (p(\boldsymbol{y}) - g_{n,d}(\boldsymbol{y}))^{0.5}\right)^2
\end{aligned}$$

Using Cauchy-Schwartz inequality, $\left(\int d\boldsymbol{y} a(\boldsymbol{y}) b(\boldsymbol{y})\right)^2 = \int d\boldsymbol{y}|a(\boldsymbol{y})|^2 \int d\boldsymbol{y}|b(\boldsymbol{y})|^2$

$$\leq \left(\int d\boldsymbol{y}| \log f(\boldsymbol{y})|^2 |p(\boldsymbol{y}) - g_{n,d}(\boldsymbol{y})|\right) \left(\int d\boldsymbol{y}|p(\boldsymbol{y}) - g_{n,d}(\boldsymbol{y})|\right)$$

The first term will be bounded by $\mathbb{E}_p \log^2 f + \mathbb{E}_g \log^2 f$ which is some finite quantity by assumption. To bound the second term, we use Thm. 1 of Lugosi and Nobel (1996) which is restated below.

Theorem 4 (Theorem 1 of Lugosi and Nobel (1996))**.** *Let* $\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)}, \ldots$ *be IID random vectors in* $\mathbb{R}^t$ *and* $\boldsymbol{y}^{(i)} \sim p(\boldsymbol{y})$*. Let* $\Pi = \{\pi_1, \pi_2 \ldots\}$ *be a fixed partitioning scheme for* $\mathbb{R}^t$ *and let* $\mathcal{A}_n$ *be the collection of all possible partitions associated with the rule* π_n*, and* $\pi_n[\boldsymbol{y}]$ *denotes the partition in which* $\boldsymbol{y}$ *lies. Define* $m(\mathcal{A}) = \max_{\pi\in\mathcal{A}} |\pi|$ *as the maximum size of partition,* $\Delta^*(\mathcal{A}, n)$ *is the maximum number of distinct partitions of any* n *points in* $\mathbb{R}^t$ *induced by* $\mathcal{A}$*. If as* $n \to \infty$*,*

- $n^{-1}m(\mathcal{A}_n) \to 0$
- $n^{-1} \log \Delta^*(\mathcal{A}_n, n) \to 0$
- $P(\{\boldsymbol{y} : diam(\pi_n[\boldsymbol{y}]) > \gamma\}) \to 0$*, i.e., the size of each bin goes to zero.*

Then the histogram density estimates, $g_{n,d}$ *are strongly consistent in* L_1*, i.e., with probability 1*

$$\int |p(\boldsymbol{y}) - g_{n,d}(\boldsymbol{y})| d\boldsymbol{y} \to 0$$

Since, we used a H dimensional binary hashcode in this work, $m(\mathcal{A}_n) \leq 2^H$. Now each bit of hashcode function can be seen as partitioning the input space (or transformed input space) by hyperplanes. We know that n points can be split by hyperplanes in n^{t^*} ways (Cover, 1965), where t^* is the effective-input dimension. Thus, $\Delta^*(\mathcal{A}_n, n) \leq n^{tH}$. We can see that the first two conditions are satisfied due to the regularity conditions.

Since input space is bounded and H increases as n increases, the diameter of bins would shrink and tend to 0 eventually. Hence,

$$\int |p(\boldsymbol{y}) - g_{n,d}(\boldsymbol{y})| d\boldsymbol{y} \to 0$$

. □

A.2 VARIANCE OF THE ESTIMATOR

Theorem 5. ***Thoerem 2 restated*** *For some data distribution p and conditional model distribution q and $-\log q(\boldsymbol{y}_t|(.)) \in [-Q, Q]$. Let $\hat{T}^q_{X\to Y}$ denote the n-sample estimate of transfer entropy. Then with probability $1-\delta$ ($\delta > 0$), we have*

$$\left|\hat{\mathcal{T}}^q_{X\to Y} - \mathcal{T}^q_{X\to Y}\right| \leq 2Q\sqrt{\frac{2}{n}\ln\frac{4}{\delta}} \tag{19}$$

Proof. To estimate transfer entropy, we use difference of empirical estimate of two conditional entropy terms, i.e,

$$\mathcal{T}^q_{X\to Y} = \mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) - \mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1})$$

Now, we can write the error as:

$$\begin{aligned}\left|\hat{\mathcal{T}}^q_{X\to Y} - \mathcal{T}^q_{X\to Y}\right| &= \left|\left(\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1})\right) - \left(\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1})\right)\right| \\ &\leq \left|\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1})\right| + \left|\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1})\right|\end{aligned}$$

We will bound the probability of error in transfer entropy estimates by bounding the error of both terms in both R.H.S terms of above expression using additive-chernoff bounds (similar to McAllester and Stratos (2020)). We can write,

$$\begin{aligned}&P\left(\left|\hat{\mathcal{T}}^q_{X\to Y} - \mathcal{T}^q_{X\to Y}\right| \leq \epsilon\right) \\ &\geq \sum_{e=0}^{\epsilon} P\left(\left|\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1})\right| \leq e\right) P\left(\left|\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1})\right| \leq \epsilon - e\right) \\ &\geq P\left(\left|\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1})\right| \leq \epsilon/2\right) P\left(\left|\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1})\right| \leq \epsilon/2\right)\end{aligned} \tag{20}$$

Consider, $\hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) = \sum_{i=1}^{n} \log q(y_t^{(i)} = c|\boldsymbol{y}_{t-1}^{(i)})$, and, $\mathbb{E}\,\hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) = \mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1})$. Since $\log q \in [-Q, Q]$, using additive chernoff bounds, we have

$$P\left(\left|\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1})\right| \geq \epsilon/2\right) \leq 2e^{-\frac{2n(\epsilon/2)^2}{(2Q)^2}} = 2e^{-\frac{n\epsilon^2}{8Q^2}}$$

and similarly,

$$P\left(\left|\mathcal{H}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1}) - \hat{\mathcal{H}}(\mathcal{Y}_t|\boldsymbol{\mathcal{Y}}_{t-1}, \boldsymbol{\mathcal{X}}_{t-1})\right| \geq \epsilon/2\right) \leq 2e^{-\frac{n\epsilon^2}{8Q^2}}$$

Substituting in 20, we have

$$P\left(\left|\hat{\mathcal{T}}^q_{X\to Y} - \mathcal{T}^q_{X\to Y}\right| \leq \epsilon\right) \leq \left(1 - 2e^{-\frac{n\epsilon^2}{8Q^2}^2}\right)^2$$

or,

$$P\left(\left|\hat{\mathcal{T}}^q_{X\to Y} - \mathcal{T}^q_{X\to Y}\right| \geq \epsilon\right) \leq 4e^{-\frac{n\epsilon^2}{8Q^2}}$$

Setting, $4e^{-\frac{n\epsilon^2}{8Q^2}} = \delta$, we get $\epsilon = 2Q\sqrt{\frac{2}{n}\ln\frac{4}{\delta}}$ □

Remark: Using above results, it becomes straight-forward to compute the bounds on mean square error of the estimator or variance as below:

Algorithm 2 Unsupervised Learning of LSH functions

Require: $\boldsymbol{Y} = (\boldsymbol{y}^{(1)}, \cdots, \boldsymbol{y}^{(n)}), H.$
for $l = 1 \rightarrow H$ **do**
  $\boldsymbol{Y}^{(l)} \leftarrow$ sampleFromHashcodeBinOfHighEntropy$(\boldsymbol{Y}, \boldsymbol{C})$
  $h_l(.) \leftarrow$optimizeSplit($\boldsymbol{Y}^{(l)}, \boldsymbol{Y}, \boldsymbol{C}$)
  $\boldsymbol{C}(:, l) \leftarrow h_l(\boldsymbol{Y}\ )$
end for
Return $h_1(.), \cdots, h_H(.), \boldsymbol{C}.$

$$
\begin{aligned}
\mathbb{E}\left(\hat{\mathcal{T}}^q_{X\rightarrow Y} - \mathcal{T}^q_{X\rightarrow Y}\right)^2 \leq & \left(2Q\sqrt{\frac{2}{n}\ln\frac{4}{\delta}}\right)^2 P\left(\left|\hat{\mathcal{T}}^q_{X\rightarrow Y} - \mathcal{T}^q_{X\rightarrow Y}\right| \leq 2Q\sqrt{\frac{2}{n}\ln\frac{4}{\delta}}\right) + \\
& P\left(\left|\hat{\mathcal{T}}^q_{X\rightarrow Y} - \mathcal{T}^q_{X\rightarrow Y}\right| \geq 2Q\sqrt{\frac{2}{n}\ln\frac{4}{\delta}}\right)(Q-(-Q))^2 \\
& = (1-\delta)\left(\frac{8Q^2}{n}\ln\frac{4}{\delta}\right) + 4Q^2\delta \\
& = 4Q^2\left((1-\delta)\frac{2}{n}\ln\frac{4}{\delta} + \delta\right)
\end{aligned}
$$

B MORE ALGORITHMIC DETAILS

In this section, we introduce more details about unsupervised learning of hash functions. The main idea behind locality sensitive hashing (LSH) is to have a set of hash functions, $\boldsymbol{h}(.) = \{h_1(.), \cdots, h_H(.)\}$, with each one, $h_l(.)$, randomly splitting the input space into two parts; $h_l(\boldsymbol{y}) \in \{0, 1\}$. Despite the randomness of an individual hash function, putting multiple hash functions together ensures that the data points belonging to same hashcode bin are similar to each other (Indyk and Motwani, 1998; Zhao et al., 2014; Wang et al., 2017).

In a data-driven LSH approach to learning a hash function $h_l(.)$, a subset of data points $\boldsymbol{Y}^{(l)}$ is subsampled from the superset of data points, $\boldsymbol{Y} = \{\boldsymbol{y}^{(1)}, \cdots, \boldsymbol{y}^{(n)}\}$, for which the hash functions (hashcodes) are optimized. A binary split of $\boldsymbol{Y}^{(l)}$ into two subsets can be generalized as a split of the entire input space (so for $\boldsymbol{Y}$) through a binary classification model, and that model is the resulting hash function $h_l(.)$ (Joly and Buisson, 2011; Kulis and Grauman, 2012; Garg et al., 2019b). Garg et al. (2019a) proposed to find the optimal split of $\boldsymbol{Y}^{(l)}$ such that it generalizes across both training and test (nearly-unsupervised setting). They also proposed to sample the subset $\boldsymbol{Y}^{(l)}$ locally from a hashcode bin so as to capture more fine-grained differences between data points with in the local neighborhoods.

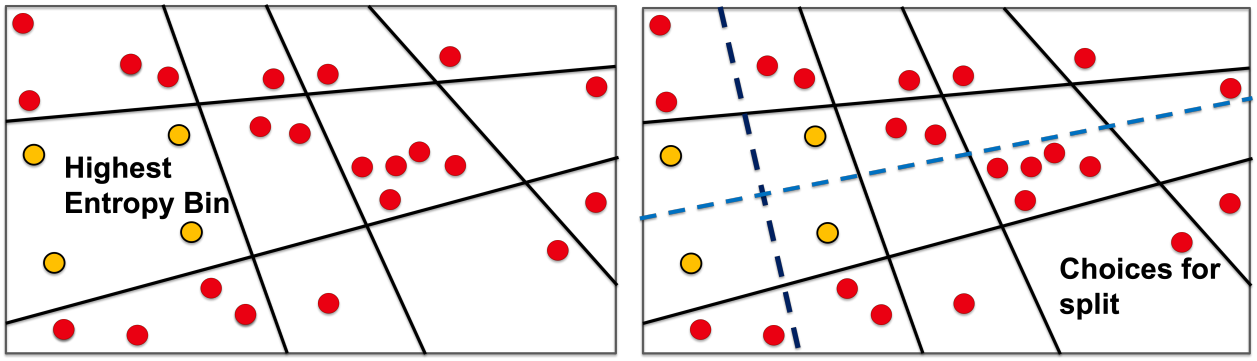
Here we extend their approach from a nearly-unsupervised setting to an unsupervised learning setting. We learn hash functions in a greedy fashion with pseudo code shown in Alg. 2. There are two key steps in the greedy approach for optimizing $h_l(.)$ (see Fig. 6 for a visual illustration). First, as per the $l-1$ number of hash functions optimized so far, we obtain hashcodes based binning of all the data points in $\boldsymbol{Y}$, then select a hashcode bin to sample $\boldsymbol{Y}^l$ from within the selected bin. If the maximal size of a bin is above a certain threshold, it suffices to just select the bin with the maximum size. Otherwise we *select the hashcode bin with the highest entropy* of data points, as shown below,

$$
\max_{\boldsymbol{c}} \mathcal{H}(\boldsymbol{\mathcal{Y}}|\boldsymbol{\mathcal{C}} = \boldsymbol{c}), \tag{21}
$$

wherein $\mathcal{H}(\boldsymbol{\mathcal{Y}}|\boldsymbol{\mathcal{C}} = \boldsymbol{c})$ is entropy of $\boldsymbol{\mathcal{Y}}$ within a particular hashcode bin $\boldsymbol{c}$; the entropy term can be computed in terms of nearest neighbor distances within the bin itself using KL estimator due to Kozachenko and Leonenko (1987).[4]

Having sampled $\boldsymbol{Y}^l$ from a hashcode bin with the highest entropy (Fig. 6(a)), the second step is to optimize its split. In Fig. 6(b), a split of the sampled points corresponds to a split of the entire 2-D input space (dashed line). Between the two choices of splits, the one with vertical dashed line (Navy colored) is optimal since it leads to better splits of the other hash bins (including self). Mathematically, we characterize this criterion of splitting the hashcode bins as,

[4]KL estimator should be accurate for empirical estimates of entropy within local neighborhoods, even in high dimensions, since a small value of k is optimal in such case.



(a) Select highest entropy bin. (b) Optimize split.

Figure 6: This figure illustrates the learning of a hash function. In 6(a) and 6(b), black lines refer to hash functions learned previously. The intersection of the lines correspond to hashcode bins, with data points (red dots) dispersed across these bins. If data points are highly dispersed within a bin, as is the case for the yellow points, it is a good candidate for sampling data points from it, so as to learn a new hash function which would split the selected bin to reduce the dispersion within it, and potentially split the other bins as well. Mathematically, dispersion of data in high dimensions can be characterized as entropy of the data distribution, see the highest entropy bin in 6(a). In 6(b), there are multiple ways of splitting the high entropy cluster (dashed lines), and the optimal choice is one which splits the other bins as well (Navy colored vertical dashed line).

$$\max_{h_l(.)} \mathcal{H}(h_l(\boldsymbol{y})|h_1(\boldsymbol{y}), \cdots, h_{l-1}(\boldsymbol{y})). \tag{22}$$

It ensures that hash function $h_l(.)$ is disentangled (not redundant) w.r.t. the hash functions optimized previously. It is cheap to compute empirically, by simply counting the fraction of ones from the output of $h_l(.)$ for each of the bins. We refine this criterion further for a hashcode bin that is of a relatively small size (number of data points in it) and has high entropy of $h_l(.)$ (less-biased proportion of ones). For any such hashcode bin $\boldsymbol{c}$, we propose to maximize the KL-divergence between the data distribution from the two partitions of the bin, emerging from $h_l(.)$:

$$\max_{h_l(.)} \mathcal{D}^c_{KL}(p(\boldsymbol{y}|h_l = 0)||p(\boldsymbol{y}|h_l = 1)). \tag{23}$$

We empirically estimate $\mathcal{D}^c_{KL}(.)$ from the ratio of the nearest neighbor distances within and across the two partitions of the bin (Zhao and Lai, 2020).

We eliminate a large fraction of candidates for a split of $\boldsymbol{Y}^l$ through constraints based on info-theoretic clustering along with a divide and conquer procedure. The algorithm is parallelizable, and is overall compute efficient.[5] While the proposed LSH algorithm has various applications, in Sec. 4, we evaluate it only for the purpose of robust estimation of transfer entropy.

C MORE DETAILS OF NEUROSCIENCE DATA ANALYSIS

The experiments used multiple high-density extracellular electrophysiology probes to simultaneously record spiking activity from a wide variety of areas in the mouse brain, ranging from the subcortical region, such as the thalamus, to multiple the visual cortices, such as primary visual cortex (V1), lateral medial visual area (LM), rostrolateral visual area (RL), anterolateral visual are (AL), anteromedial visual area (AM), etc. The neural activities were recorded while the animals were head-fixed and were passively presented with visual stimuli. The details of the experimental setup can be found in (Siegle et al., 2021; Institute, 2020). We used animal with session-id `798911424`. One experiment contained a mixture of many stimulus types, such as natural movies, flashes, Gabor filters, drifting gratings, etc. We selected recording trials of drifting gratings as they could strongly elicit neural responses. The visual stimulus of each trial lasted for two seconds with one-second rest in between without any visual stimulus. We randomly subsampled 100 trials without replacement from 13 conditions of drifting gratings with 15 repeated trials each, so totally out of 195 trials. The condition ids are: `275`, `268`, `270`, `284`, `274`, `249`, `261`, `278`, `280`, `256`, `260`, `257`, `281`. The first 500 ms of each trial after stimulus onset was extracted, as the early visual activities mainly involved feedforward interactions, thus could better reflect the hierarchy of the visual system. The number of neurons in visual cortical areas recorded by one probe was roughly around 100. The raw data was composed of sequences of action potential timestamps or counting processes of each neuron. We time-binned the timestamps with 0.1 ms resolution, then averaged the time series across all neurons for each brain region. The time lags of all TE estimators are all 20 ms, with 40 time steps and the length of each time step is 0.5 ms.

[5]Code will be released upon publication.

Here are more details of the main Fig. 4. The hierarchical order of the brain regions, from low to high, is V1, LM, RL, AL, and AM (Harris et al., 2019; Siegle et al., 2021). The columns indicate the source regions and the rows indicate the target regions. We sort the regions according to the hierarchical order from left to right and top to bottom. If $\mathcal{T}_{\mathrm{A}\to\mathrm{B}} > 0$ only if the hierarchical order of A is greater than B, then positive TE values will stay under the diagonal. Fig. 4(a) shows the TE between regions. A larger value means the source region contributes more significantly to the target region's entropy, which implies the direction of information flow and the source region has an impact on the activity of the downstream target region. All large values concentrate in the lower left triangle, which means the low-order regions impact the high-order regions, thus the conclusions agree with the hierarchical order found by other anatomical or functional methods. Importantly, the TE values in each row has a decreasing trend, so for one target region, the lower-order area contributes more than the higher-order. However, interestingly, the TEs in columns do not have an obvious monotonic pattern. For example, the V1 column is not in decreasing order from top to the bottom, which means V1 does not influence more on the higher-order region. $\mathcal{T}_{\mathrm{V1}\to\mathrm{RL}} > \mathcal{T}_{\mathrm{V1}\to\mathrm{AM}}$ but AM has higher hierarchical order than RL. This observation also applies to LM and RL. It is worth noting that one area having higher hierarchical order than another does not necessarily indicate non-zero or large TE. For example, AL's order is larger than RL, but $\mathcal{T}_{\mathrm{RL}\to\mathrm{AL}} = 0$. The top right triangle matrix has almost all zeros except for two small positive values, $\mathcal{T}_{\mathrm{AL}\to\mathrm{LM}}$ and $\mathcal{T}_{\mathrm{AM}\to\mathrm{RL}}$. We purposely selected the early visual activities to eliminate the influences of the following feedback interactions. This is why the results do not show strong interaction from high-order regions to low-order regions. But this does not fully remove the effects. Feedback information flow AL $\to$ LM and AM $\to$ RL may occur earlier than 500 ms. If we enlarge the time window from 500 ms to 2000 ms to cover the entire trial window, the upper triangle matrix will show more positive values (figures or results not shown). As the current neuroscience research is still not clear when and how the feedback signal intertwines with the feedforward signal, we bypass the analysis of these complicated patterns, but our method suggests the existence of strong feedback loops, and it is interesting to notice that the feedback propagation could start from high-order regions.

In contrast, other methods do not properly reveal the hierarchical relationships among the visual areas. The TE values of the kNN method in Fig. 4(b) show an almost symmetric matrix, that hardly provides any directional information between brain regions. This issue is also observed in Fig. 3, where the kNN method yields almost the same positive value no matter the true value is close to zero or one. The Copent method in Fig. 4(c) falsely indicates that most higher-order regions impact V1's activity, the lowest order area. It also ignores the interactions between LM, RL, and AL, where most corresponding TE values are nearly zeros. The ITENE method in Fig. 4(d) also suffers from this issue showing many opposite hierarchical relations between visual areas where many large values concentrate in the top right triangle matrix instead of the bottom left matrix.