Renormalized Mutual Information for Artificial Scientific Discovery

Leopoldo Sarra,[1,*] Andrea Aiello,[1] and Florian Marquardt[1,2]

[1] *Max Planck Institute for the Science of Light, Erlangen, Germany*
[2] *Department of Physics, Friedrich-Alexander Universität Erlangen-Nürnberg, Germany*

(Dated: June 5, 2020)

We derive a well-defined renormalized version of mutual information that allows to estimate the dependence between continuous random variables in the important case when one is deterministically dependent on the other. This is the situation relevant for feature extraction, where the goal is to produce a low-dimensional effective description of a high-dimensional system. Our approach enables the discovery of collective variables in physical systems, thus adding to the toolbox of artificial scientific discovery, while also aiding the analysis of information flow in artificial neural networks.

Introduction. – One of the most useful general concepts in the analysis of physical systems is the notion of collective coordinates. In many cases, ranging from statistical physics to hydrodynamics, the description of a complex many-particle system can be dramatically simplified by considering only a few collective variables like the center of mass, an order parameter, a flow field, or vortex positions. However, when encountering new situations, it is not clear a priori which low-dimensional "feature" $y = f(x)$ is best suited as a compact description of the high-dimensional data x. This is the domain of unsupervised feature extraction in computer science, where large datasets like images or time series are to be analyzed [1]. Future frameworks of artificial scientific discovery [2–5] will have to rely on general approaches like this, adding to the rapidly developing toolbox of machine learning for physics [6–8].

In assessing the quality of a proposed feature, mutual information provides a general approach [9, 10]. In general, the mutual information $I(x, y)$ answers the following question: if two random variables y and x are dependent on one another, and we are provided with the value of y, how much do we learn about x? Technically, it is defined via $I(x, y) = I(y, x) = H(y) - H(y|x)$, where $H(y|x)$ is the conditional entropy of y given x [10]. Maximization of mutual information can be used to extract "optimal" features [11], as sketched in Fig. 1.

There exists, however, a well-known important problem in evaluating the mutual information for *continuous* variables with a *deterministic* dependence [12, 13], which is exactly the case relevant for feature extraction. In this case, $I(x, y)$ diverges, and it is not clear how to properly cure this divergence without losing important properties of I. Specifically, reparametrization invariance turns out to be crucial: applying a bijective function to obtain $y' = g(y)$ does not change the information content, and thus $I(x, y') = I(x, y)$.

In this work, we introduce a properly *renormalized* version of mutual information for the important case of feature extraction with continuous variables:

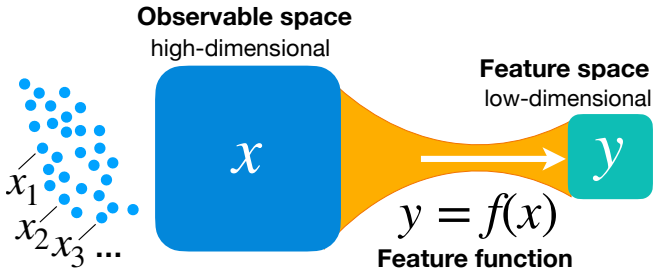

Figure 1. Feature extraction, where a high-dimensional "microscopic" description x (such as the configuration of a many-particle system) is mapped to a low-dimensional feature $y = f(x)$. This is the case where the renormalized mutual information presented in this article is needed for feature optimization.

$$\tilde{I}(x, y) = H(y) - \int dx P_x(x) \ln \sqrt{|\nabla f(x) \cdot \nabla f(x)|}, \quad (1)$$

where $x \in \mathbb{R}^N$, $y = f(x) \in \mathbb{R}^K$; we use $|\dots|$ as a short-hand notation for the determinant, as well as $\nabla f(x) \cdot \nabla f(x)$ for $(\sum_i \partial_i f_\mu \partial_i f_\nu)_{\mu\nu}$, with $1 \leq i \leq N$ and $1 \leq \mu, \nu \leq K$, i.e. the $K \times K$ matrix resulting from the product of the $(K \times N)$ Jacobian matrix $\nabla f(x)$ and its transpose. The quantity $\tilde{I}$ is well-defined and finite. In addition, it preserves fundamental properties of mutual information – among which the invariance under reparametrization of the features:

$$\tilde{I}(x, g(y)) = \tilde{I}(x, y). \quad (2)$$

for a bijective function $g : \mathbb{R}^K \to \mathbb{R}^K$. We will derive and discuss below the meaning and usefulness of the renormalized quantity $\tilde{I}$.

Mutual Information is used in many cutting edge machine learning applications, helping to improve the intermediate layers of a neural network [14, 15], to increase the interpretability of Generative Adversarial Networks [16], to analyze the behavior of neural networks during training [17, 18] through the Information Bottleneck method [19, 20], and for feature extraction via mutual information optimization [21]. It can be also used to characterize the variables in a renormalization group procedure

* leopoldo.sarra@mpl.mpg.de

[22]. Practical estimation of mutual information is a non-trivial task [23], but the recent development of carefully designed bounds [24] permitted to use neural networks to perform the estimation [25], making it feasible also in high-dimensional spaces.

However, there is a problem when dealing with deterministically-dependent continuous features: the conditional entropy $H(y|x)$ formally diverges as $-\log\delta(0)$ whenever y is a deterministic function of x. To understand why, it is enough to take its definition, $H(y|x) = -\int dxdyP_x(x)P(y|x)\ln P(y|x)$, and plug in $P(y|x) = \delta(y - f(x))$. This is specific to continuous variables: for discrete random variables conditional entropy would be zero in this case and mutual information would coincide with the entropy of one of the variables. Therefore, it is clear that, to deal with a deterministic continuous dependence, it is necessary to somehow redefine mutual information. Past remedies involved adding noise to the feature y or (equivalently) to simply consider the non-diverging term $H(y)$ [21, 26], as briefly suggested in the seminal InfoMax paper [11]. However, those remedies lead to a very undesireable property: they break the fundamental reparametrization invariance of mutual information. In this scheme, when features are unconstrained, any two features (with non-zero variance) can be made to have the same entropy $H(y)$ simply by rescaling. Thus, in the context of feature optimization, they would be considered equally favorable, even if they represent very different information about the high-dimensional data x. The reason is that such a scheme completely ignores the diverging quantity $H(y|x)$. In contrast, we show that $H(y|x)$ contains a non-trivial finite dependence on the feature $f(x)$, which must be taken into account to obtain consistent results.

Renormalized Mutual Information. – In any physical system, there are small pre-existing measurement uncertainties associated with extracting the microscopic observables x. Thus, loosely speaking, when trying to deduce information about x given the value of y, we have to be content with resolving x up to some spread ε. Motivated by this, we first consider a finite regularized quantity $I_\varepsilon(x, y)$. It is defined as the mutual information between the observable x and the feature function applied to a noisy version of the observable: $y = f(x + \varepsilon\lambda)$, where $\varepsilon \in \mathbb{R}$ is the noise strength and $\lambda \in \mathbb{R}^N$ is a random multidimensional Gaussian of zero mean and unit covariance matrix. In the limit $\varepsilon \to 0$ we recover the original definition of mutual information, which diverges logarithmically. Even in that limit, the nature of the adopted noise distribution (e.g. isotropy, independence of x) still matters, and it corresponds to imposing some hypothesis about the observed quantities x (e.g. same measurement uncertainty in all variables). We discuss these generalizations at the end of this work.

Consider

$$P(y|x) = \int d\lambda P_\lambda(\lambda)\delta(y - f(x + \varepsilon\lambda)). \tag{3}$$

When $\varepsilon \ll 1$, we can expand $f(x+\varepsilon\lambda) \simeq f(x)+\varepsilon\lambda\cdot\nabla f(x)$. By explicit calculation, it can be easily found that $P(y|x)$ is a Gaussian distribution of zero mean and covariance matrix $\varepsilon^2\nabla f(x)\cdot\nabla f(x) = \varepsilon^2\left(\sum_i \partial_i f_\mu \partial_i f_\nu\right)_{\mu\nu}$. We can now calculate the conditional entropy and get

$$H(y|x) = \int dx P_x(x)\ln\sqrt{|\nabla f(x)\cdot\nabla f(x)|} + KH_\varepsilon \tag{4}$$

where H_ε is the entropy of a one-dimensional Gaussian with variance ε^2. We see that the first term only depends on the features, and the second only depends on the noise variable. Only this term diverges when $\varepsilon \to 0$. Therefore

$$\tilde{I}_\varepsilon(x, y) = I_\varepsilon(x, y) + KH_\varepsilon \tag{5}$$

has a well defined limit $\varepsilon \to 0$ and still contains all the dependence on $f(x)$. By performing the limit we obtain our main result, Eq. (1).

We can easily show that Eq. (1) is invariant under feature reparametrization. Consider an invertible function $z = g(y) : \mathbb{R}^K \to \mathbb{R}^K$. We can rewrite the entropy of z as the entropy of y plus an extra term, which cancels with the term that appears by differentiating $\ln|\nabla g(f(x))|$, leading to Eq. (2). We emphasize the importance of this property: after an invertible transformation on the variable y, no information should be lost, and the new variable should have the same mutual information with x as the old one. In contrast, if one were to regularize by adding Gaussian noise η to the feature y instead of to x, i.e. $y = f(x) + \varepsilon\eta$, the final result would depend on the feature only via $H(y)$. Reparametrization invariance would not hold anymore under this alternative regularization: we have $I_\varepsilon(x, g(f(x) + \varepsilon\eta)) = I_\varepsilon(x, f(x) + \varepsilon\eta)$ but not $I_\varepsilon(x, g(f(x)) + \varepsilon\eta) = I_\varepsilon(x, f(x) + \varepsilon\eta)$ as Eq. (2) would require.

The small price to pay for making mutual information between two deterministically-dependent variables finite is that when there is no dependence, e.g. $y = \text{const.}$, we don't get zero anymore but $-\infty$ (similar to the entropy of a delta distribution). In addition, given the different roles that x and y play, renormalized mutual information is not symmetric anymore in its arguments. From a different perspective, Eq. (1) can be expressed as a particular kind of *Information Loss* [27, 28] [29].

Mutual information obeys inequalities like $I(x, (y_1, y_2)) \geq I(x, y_1)$, which translate to the regularized version I_ε. However, naively taking $\varepsilon \to 0$ results in an empty inequality $\tilde{I}(x, (y_1, y_2)) + \infty \geq \tilde{I}(x, y_1)$. By contrast, starting from $I(x, (y_1, y_2)) \geq I(x, y_1) + I(x, y_2) - I(y_1, y_2)$, we can take the same limit and obtain a useful finite result:

$$\tilde{I}(x, (y_1, y_2)) \geq \tilde{I}(x, y_1) + \tilde{I}(x, y_2) - I(y_1, y_2). \tag{6}$$

In the special case where the dimensions of y_1 and y_2 add up to the dimension of x, and where the mapping $x \mapsto (y_1, y_2)$ is bijective, reparametrization invariance produces $\tilde{I}(x, (y_1, y_2)) = \tilde{I}(x, x) = H(x)$, such that we find

$$H(x) \geq \tilde{I}(x, y_1) + \tilde{I}(x, y_2) - I(y_1, y_2)\,. \qquad (7)$$

If one constructs y_2 to be independent of y_1, the third term on the right-hand-side vanishes as well. However, it would be impermissible to drop $\tilde{I}(x, y_2)$, since it can have any sign.

Feature comparison. – In practice, the observable x might be a high-dimensional vector describing the generalized coordinates of a many-particle system or the values of a fluctuating field on a lattice. The renormalized mutual information can be used to find out how useful any given "macroscopic" quantity (i.e. a feature $y = f(x)$) would be in characterizing the system. The result depends on the statistical distribution of x. It might be the Boltzmann distribution in equilibrium or a distribution of "snapshots" of the system configuration during some arbitrary time evolution. When control parameters such as temperature or external fields change the distribution of x, the optimal feature can change. Intuitively, observing a feature with higher $\tilde{I}$ is more effective in narrowing down the set of underlying configurations x compatible with the observed value, thus yielding more descriptive power about the system.

In Fig. 2, we illustrate these ideas in a straightforward model. We consider a fluctuating 1D field on a lattice and place a wavepacket of fixed shape at a random position. For comparison, we evaluate $\tilde{I}$ for a variety of features. Because of reparametrization invariance (Eq. (2)), the scaling of any of them is irrelevant, as is any bijective nonlinear transformation. In this example, we find there is a sharp transition, as a function of noise strength, in the choice of the "best" feature (within the given set of features). Depending on the regime, different one-dimensional features with clearly distinct physical meaning become optimal. Among the features, we also consider Principal Component Analysis (PCA) [30], which is the simplest and most known algorithm for *linear* feature extraction. In our context, PCA corresponds to a feature $f(x) = \sum_j x_j u_j$, where u is the eigenvector associated to the largest eigenvalue of the covariance matrix $\langle x_i x_j \rangle - \langle x_i \rangle \langle x_j \rangle$. Several of the considered features are nonlinear in x and, as Fig. 2b shows, one of them yields a higher $\tilde{I}$ than PCA. In Fig. 2c,d, we show another example, this time analyzing two-dimensional features: a many-particle system forming drops of fluctuating shapes. Again, the inspection of $\tilde{I}$ reveals when a given feature is more descriptive.

Unconstrained feature optimization. – Instead of comparing different plausible features, we can also consider a whole class of parametrized features and optimize $\tilde{I}$ over the parameters to get the best feature. Many parametrizations of $f(x)$ are possible, but for generality we opted for a multilayer neural network [31], where $f(x) = f_\theta(x)$ with θ representing the weights and biases of the network. This allows to represent arbitrary features (given sufficiently many neurons). Then, gradient ascent on $\tilde{I}(x, f_\theta(x))$ can be implemented easily, e.g.

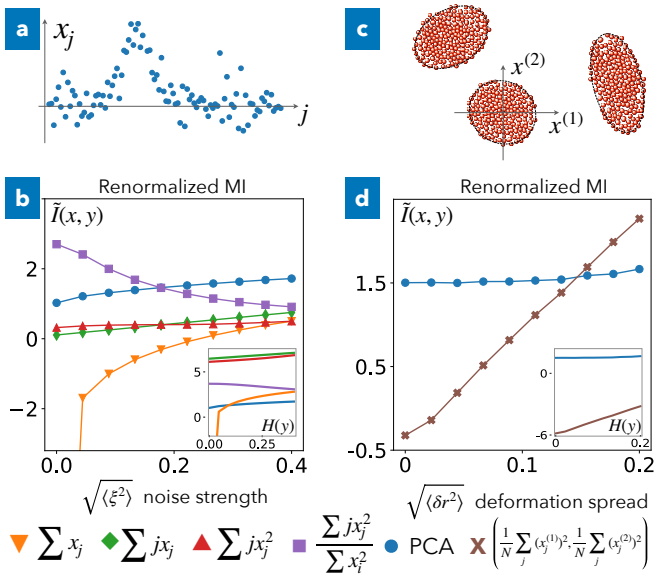

Figure 2. Comparison of features using the renormalized mutual information $\tilde{I}$, for examples with high-dimensional data x. (a) Example: Fluctuating 1D field on a lattice, with a randomly placed "wave packet" (we depict one single random sample). (b) $\tilde{I}$ for several features, as a function of the size of the field fluctuations. We consider as possible features: the average field $f(x) = \frac{1}{N}\sum_{j=1}^{N} x_j$, the position j weighted by the field amplitude, $\frac{1}{N}\sum_{j=1}^{N} j x_j$, or weighted by the field intensity, $\frac{1}{N}\sum_{j=1}^{N} j x_j^2$, as well as the "normalized" feature $\sum_{j=1}^{N} j x_j^2 / \sum_{i=1}^{N} x_i^2$ (similar to an expectation value in quantum mechanics) and the principal component analysis (PCA) with one component [$x_j = \xi_j + e^{-(j-\bar{\jmath})^2/\delta j^2}$ with ξ_j i.i.d. Gaussian random variables, $\delta j = 9$, $\bar{\jmath}$ uniformly random $\in [30, 70]$, $N = 100$. We plot $\tilde{I}$ vs. $\sqrt{\langle \xi_j^2 \rangle}$]. (c) Example: Two-dimensional "drops" with elliptical shapes of fixed area but with fluctuating deformation amplitude δr and orientation. Initially randomly placed particles were thermalized using a generalized Lennard-Jones type interaction potential, constrained by the shape of the drop (we depict three random samples with different deformation and orientation). (d) Renormalized mutual information vs. deformation spread $\sqrt{\langle \delta r^2 \rangle}$ for two-dimensional features $f(x)$: the two best linear features according to PCA, and a nonlinear feature sensitive to shape deformations, $f(x) = (\frac{1}{N}\sum_j (x_j^{(1)})^2, \frac{1}{N}\sum_j (x_j^{(2)})^2)$. In the insets, we show the entropy $H(f(x))$ for the same features. Note that this quantity is not reparametrization invariant.

with TensorFlow [32]. In Fig. 3 we show the optimization of a nonlinear one-dimensional feature for a non-Gaussian two-dimensional distribution. The convergence of the feature is best in the regions of highest probability weight (where getting the feature right is important), while low-probability regions are less reliable (but also less relevant). Note that the feature space has an extra degree of freedom, given by invertible endomorphisms in that space that leave renormalized mutual information unchanged. This can be exploited to enforce additional

constraints on the extracted features. Before comparing different features, for example obtained through the training of different neural networks, it is important to first "normalize" them. This can be done also after the optimal feature has been found.

The use of a neural network for a low-dimensional feature $f_\theta(x)$ is directly suitable, in the manner described above, also for a high-dimensional x-space (e.g. given by images or configurations of many-particle systems). Furthermore, the estimation of regularized mutual information is more challenging, but still feasible also with feature with a larger dimension. To evaluate the second term in Eq. (1), one can rely on statistical sampling of x. Indeed, one can immediately obtain the required ∇f, since neural networks are differentiable functions. More care is needed to efficiently approximate the first term of Eq. (1), $H(y)$. We have no access to the functional form of $P_y(y)$, but we can sample from it (since $y = f(x)$). For example, one can introduce a reference distribution $P_r(y)$ and rewrite $H(y) = -\langle \ln P_r(y)\rangle_{y\sim P_y(y)} - KL(P_y||P_r)$, where the last term is the Kullback-Leibler divergence [31] between $P_r(y)$ and $P_y(y)$. This term can be estimated by means of adversarial techniques [33], i.e. by optimizing over an auxiliary "discriminator" neural network $D(y)$:

$$H(y) = -\langle \ln P_r(y)\rangle_{y\sim P_y(y)} + \\ + \min_D \left(\langle D(y)\rangle_{y\sim P_y(y)} + \langle e^{-D(y)}\rangle_{y\sim P_r(y)} - 1 \right). \quad (8)$$

For convenience, $P_r(y)$ could be chosen as a Gaussian distribution. This is related to the technique proposed in [25] to estimate mutual information, while properly allowing for the renormalization discussed here.

In the examples above, all the components x_j had the same physical meaning (e.g. particle coordinates). For components with different dimensions (e.g. positions and momenta), one clearly needs to decide how to compare fluctuations along these different components. To deal with that, a slight change in the regularization procedure is required. Most generally, we can consider an arbitrary distribution of the regularizing noise λ, with a covariance matrix $\Sigma(x)$ of the noise distribution $P(\lambda|x)$. This can also be x-dependent, to allow for a location-dependent "resolution". We find that it is only necessary to replace the matrix $\nabla f(x) \cdot \nabla f(x)$ in Eq. (1) with $\nabla f(x)\Sigma(x)\nabla f(x)$, thus effectively introducing a metric on x-space [34]. Note that this changes the inequality mentioned above (Eq. (7)).

Outlook. – Renormalized mutual information can be useful in many areas of statistical analysis, machine learning, and physics.

It can be directly applied in the most diverse physical scenarios. Also, many interesting variations and extensions are possible. In statistical physics, it is reasonable to expect that different phases of matter yield different optimal features. Moreover, one could optimize for feature *fields* (order parameter fields) by using convolutional layers in the neural network that parametrizes the features. The locations of defects like domain walls and vortices could be discovered as relevant features.

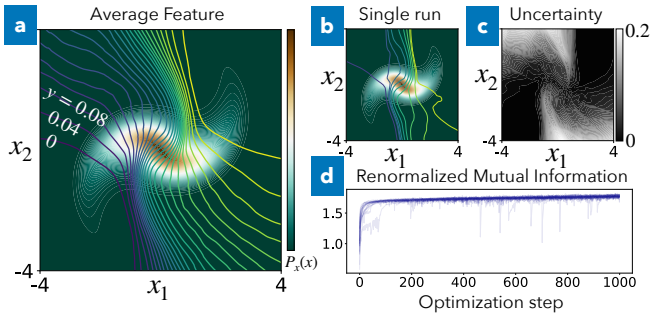

Figure 3. Unconstrained feature optimization for a 2D non-Gaussian distribution. (a) Result of the numerical optimization of the renormalized mutual information $\tilde{I}$. The 1D feature $y = f(x_1, x_2)$, shown as contour lines atop the distribution $P_x(x)$, is parametrized with a fully-connected neural network with 150 neurons for each of the 3 hidden layers. Here we average over the results from 20 independent optimization runs (each suitably normalized) and show the mean resulting feature. We "normalized" the features obtained in independent runs by applying an invertible transformation $\hat{y} = \int_{-\infty}^{y} dy' P_y(y')$ to the output of the neural network so that $\hat{y}$ always has a uniform distribution between 0 and 1. (b) Normalized feature for a single run. (c) Standard deviation of the features obtained in several independent optimization runs. (d) Renormalized mutual information during the optimization steps for many runs. [The distribution is obtained using $x_1 = x_1' \cos\alpha r - x_2' \sin\alpha r$ and $x_2 = x_1' \sin\alpha r + x_2' \cos\alpha r$, where x_1' and x_2' are Gaussian random variables, $r = \sqrt{x_1^2 + x_2^2}$, and α a fixed parameter.]

In general, an optimized low-dimensional description of a high-dimensional system can be used to make partial predictions for the time evolution. In dynamical systems, particularly in chaotic ones, the renormalized mutual information could help to discover the remaining underlying regularities of the system. Even in the presence of chaos, the evolution of collective variables can be predictable (and still non-trivial).

Quantum mechanical systems could be analyzed as well, e.g. by sampling configurations x according to a many-body state, or sampling parameters in the Hamiltonian and looking at the expectation values x of a set of commuting observables in the corresponding ground state.

Renormalized mutual information can be used as a tool to analyze deterministic representations of a dataset. Here we illustrated the approach only in settings with a very low-dimensional feature space (i.e. we considered at most at two-dimensional features), but it should be feasible to efficiently evaluate $\tilde{I}$ also with high-dimensional feature spaces, for example, starting from Eq. (8). This approach could be used to study the behavior of a neural network from an information-theoretic perspective, for example by analyzing the renormalized mutual information between the input and an intermediate layer of a

neural network during its training. This could be helpful for concepts like the "information bottleneck" [19, 35], which is known to be affected by the problems we discussed. Moreover, the important challenge of finding an optimized general representation of a high-dimensional dataset (like images) can benefit: the approach we presented provides a general-purpose representation that is purely defined by its information content (and not by the capability to accomplish arbitrarily selected tasks). Therefore, it could hopefully be useful in transfer learning scenarios, in which many classifiers can be built starting from the same high-level representation. We emphasize that the method advocated here is especially useful in cases where the dimensionality is so drastically reduced that autoencoders of any kind [36–38] would not plausibly work, since it would be impossible to find a decoder that produces an approximation of the input based on so few latent variables. This is precisely the situation important for collective variables and similar strongly reduced descriptions.

The code to calculate renormalized mutual information with some examples is publicly available [39].

Acknowledgements. – We thank Andreas Maier for discussions.

[1] Y. Bengio, A. Courville, and P. Vincent, Representation Learning: A Review and New Perspectives, IEEE Trans. Pattern Anal. Mach. Intell. **35**, 1798 (2013).

[2] R. D. King, J. Rowland, S. G. Oliver, M. Young, W. Aubrey, E. Byrne, M. Liakata, M. Markham, P. Pir, L. N. Soldatova, A. Sparkes, K. E. Whelan, and A. Clare, The Automation of Science, Science **324**, 85 (2009).

[3] M. Schmidt and H. Lipson, Distilling Free-Form Natural Laws from Experimental Data, Science **324**, 81 (2009).

[4] T. Wu and M. Tegmark, Toward an artificial intelligence physicist for unsupervised learning, Phys. Rev. E **100**, 033311 (2019).

[5] R. Iten, T. Metger, H. Wilming, L. del Rio, and R. Renner, Discovering Physical Concepts with Neural Networks, Phys. Rev. Lett. **124**, 010508 (2020).

[6] V. Dunjko and H. J. Briegel, Machine learning & artificial intelligence in the quantum domain: A review of recent progress, Rep. Prog. Phys. **81**, 074001 (2018).

[7] P. Mehta, M. Bukov, C.-H. Wang, A. G. Day, C. Richardson, C. K. Fisher, and D. J. Schwab, A high-bias, low-variance introduction to Machine Learning for physicists, Phys. Rep. **810**, 1 (2019).

[8] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, Rev. Mod. Phys. **91** (2019).

[9] T. Cover and J. A. Thomas, *Elements of information theory*, 2nd ed. (Wiley-Interscience, Hoboken, N.J, 2006).

[10] A. Papoulis and S. U. Pillai, *Probability, random variables, and stochastic processes*, 4th ed. (McGraw-Hill, Boston, Mass., 2009).

[11] A. J. Bell and T. J. Sejnowski, An Information-Maximization Approach to Blind Separation and Blind Deconvolution, Neural Comput. **7**, 1129 (1995).

[12] R. A. Amjad and B. C. Geiger, Learning Representations for Neural Network-Based Classification Using the Information Bottleneck Principle, IEEE Trans. Pattern Anal. Mach. Intell. , 1 (2019).

[13] A. Kolchinsky, B. D. Tracey, and S. V. Kuyk, Caveats for information bottleneck in deterministic scenarios, in *International Conference on Learning Representations* (2019).

[14] R. D. Hjelm, A. Fedorov, S. Lavoie-Marchildon, K. Grewal, P. Bachman, A. Trischler, and Y. Bengio, Learning deep representations by mutual information estimation and maximization, arXiv:1808.06670 [cs, stat] (2018).

[15] M. Tschannen, J. Djolonga, P. K. Rubenstein, S. Gelly, and M. Lucic, On Mutual Information Maximization for Representation Learning, in *International Conference on Learning Representations* (2020).

[16] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets, in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS16 (Curran Associates Inc., Red Hook, NY, USA, 2016) pp. 2180–2188.

[17] R. Shwartz-Ziv and N. Tishby, Opening the Black Box of Deep Neural Networks via Information, arXiv:1703.00810 [cs] (2017).

[18] A. M. Saxe, Y. Bansal, J. Dapello, M. Advani, A. Kolchinsky, B. D. Tracey, and D. D. Cox, On the information bottleneck theory of deep learning, J. Stat. Mech: Theory Exp. **2019**, 124020 (2019).

[19] N. Tishby, F. C. Pereira, and W. Bialek, The information bottleneck method, in *Proc. of the 37-th Annual Allerton Conference on Communication, Control and Computing* (1999) pp. 368–377.

[20] M. Gabrié, A. Manoel, C. Luneau, J. Barbier, N. Macris, F. Krzakala, and L. Zdeborová, Entropy and mutual information in models of deep neural networks, J. Stat. Mech: Theory Exp. **2019**, 124014 (2019).

[21] S. S. Haykin, *Neural networks: A comprehensive foundation*, 2nd ed. (Prentice Hall, Upper Saddle River, N.J, 1999).

[22] M. Koch-Janusz and Z. Ringel, Mutual information, neural networks and the renormalization group, Nat. Phys. **14**, 578 (2018).

[23] A. Kraskov, H. Stögbauer, and P. Grassberger, Estimating mutual information, Phys. Rev. E **69**, 066138 (2004).

[24] B. Poole, S. Ozair, A. Van Den Oord, A. Alemi, and G. Tucker, On Variational Bounds of Mutual Information, in *Proceedings of the 36th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 97, edited by K. Chaudhuri and R. Salakhutdinov (PMLR, Long Beach, California, USA, 2019) pp. 5171–5180.

[25] M. I. Belghazi, A. Baratin, S. Rajeshwar, S. Ozair, Y. Bengio, A. Courville, and D. Hjelm, Mutual Information Neural Estimation, in *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research, Vol. 80, edited by J. Dy

and A. Krause (PMLR, Stockholmsmässan, Stockholm Sweden, 2018) pp. 531–540.
[26] G. Deco and D. Obradovic, *An Information-Theoretic Approach to Neural Computing*, edited by J. Taylor and C. Mannion, Perspectives in Neural Computing (Springer New York, New York, NY, 1996).
[27] B. C. Geiger and G. Kubin, On the Information Loss in Memoryless Systems: The Multivariate Case, in *Proc. Int. Zurich Seminar on Communications* (2011) pp. 32–35.
[28] B. C. Geiger, C. Feldbauer, and G. Kubin, Information loss in static nonlinearities, in *2011 8th International Symposium on Wireless Communication Systems* (IEEE, Aachen, Germany, 2011) pp. 799–803.
[29] See Appendix B for the equation of Renormalized Mutual Information in terms of information loss.
[30] I. T. Jolliffe and J. Cadima, Principal component analysis: A review and recent developments, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **374**, 20150202 (2016).
[31] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016).
[32] M. Abadi *et al.*, *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems* (2015).
[33] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, in *Advances in Neural Information Processing Systems 27*, edited by Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger (Curran Associates, Inc., 2014) pp. 2672–2680.
[34] See Appendix A for the derivation of the general case with position-dependent noise.
[35] D. Strouse and D. J. Schwab, The Deterministic Information Bottleneck, Neural Comput. **29**, 1611 (2017).
[36] G. Hinton, Reducing the Dimensionality of Data with Neural Networks, Science **313**, 504 (2006).
[37] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, Extracting and Composing Robust Features with Denoising Autoencoders, in *Proceedings of the 25th International Conference on Machine Learning*, ICML '08 (Association for Computing Machinery, New York, NY, USA, 2008) pp. 1096–1103.
[38] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, Contractive Auto-Encoders: Explicit Invariance during Feature Extraction, in *Proceedings of the 28th International Conference on International Conference on Machine Learning*, ICML'11 (Omnipress, Madison, WI, USA, 2011) pp. 833–840.
[39] https://github.com/lsarra/rmi.

Appendix A: Derivation of Renormalized Mutual Information for the general case of position-dependent noise

In this section, we derive the renormalized mutual information equation,

$$\tilde{I}(x,y) = H(y) - \int dx P_x(x) \ln \sqrt{|\nabla f(x) \cdot \nabla f(x)|}, \tag{A1}$$

in the general case in which the regularizing noise also depends on x. We consider the observable distribution $x \sim P_x(x)$, with $x \in \mathbb{R}^N$. Let λ be the input-noise variable. It has a zero-mean gaussian distribution with covariance matrix $\Sigma(x)$. In the case in which we have no assumptions on the observables, we can just choose $\Sigma(x) = \mathbb{I}_N$. Let $\varepsilon \in \mathbb{R}$ represent the strength of the noise. At the end of the calculation, we want to perform the limit $\varepsilon \to 0$. First of all, we define renormalized feature

$$y = f(x + \varepsilon\lambda).$$

Its probability distribution is given by

$$P_y(y) = \int dx P_x(x) d\lambda P_\lambda(\lambda|x) \delta(y - f(x + \varepsilon\lambda)).$$

By definition, $P_\lambda(\lambda|x)$ is a gaussian distribution with zero mean. The contribution of large values of λ in the δ-function are suppressed by the factor $P_\lambda(\lambda|x)$. As a consequence, when $\varepsilon \approx 0$, we can consider the expansion of the feature function, $f(x + \varepsilon\lambda) \approx f(x) + \varepsilon \nabla f(x) \cdot \lambda$. We employ the Fourier representation of the δ-function

$$\delta(y) = \frac{1}{(2\pi)^k} \int ds e^{isy}$$

and plug in the expression of the distribution of the noise,

$$P(\lambda|x) = \frac{1}{\sqrt{(2\pi)^N |\Sigma(x)|}} e^{-\frac{1}{2}\lambda\Sigma(x)^{-1}\lambda}.$$

We get

$$P(y|x) = \int \frac{ds}{(2\pi)^k} e^{is(y-f(x))} \int \frac{d\lambda}{\sqrt{(2\pi)^N |\Sigma(x)|}} e^{-\frac{1}{2}\lambda\Sigma(x)^{-1}\lambda - is\varepsilon\nabla f(x)\cdot\lambda} = \int \frac{ds}{(2\pi)^k} e^{-\frac{\varepsilon^2}{2} s(\nabla f(x)\Sigma(x)\nabla f(x))s + i(y-f(x))s}.$$

Now, we can also perform the Gaussian integral in s and get

$$P(y|x) = \frac{1}{\sqrt{(2\pi\varepsilon)^k |\nabla f(x)\Sigma(x)\nabla f(x)|}} e^{-\frac{1}{2\varepsilon^2}(y-f(x))(\nabla f(x)\Sigma(x)\nabla f(x))^{-1}(y-f(x))}.$$

This is a Gaussian distribution with mean $f(x)$ and covariance matrix $\varepsilon\nabla f(x)\Sigma(x)\nabla f(x)$. By explicit calculation, the conditional entropy $H(y|x)$ is given by

$$H(y|x) = -\int dxdy P_x(x)P_y(y|x)\ln P(y|x) = \frac{K}{2}\ln 2\pi e\varepsilon^2 + \frac{1}{2}\int dx P_x(x)\ln|\nabla f(x)\Sigma(x)\nabla f(x)|.$$

We define

$$\tilde{I}(x,y) = \lim_{\varepsilon\to 0}\left[H(y) - H(y|x) + KH_\varepsilon\right] = H(y) - \int dx P_x(x)\ln\sqrt{|\nabla f(x)\Sigma(x)\nabla f(x)|},$$

with $H_\varepsilon = \frac{1}{2}\ln 2\pi e\varepsilon^2$. This equation is more general than Eq. (A1) and it reduces to it if we consider an isotropic noise matrix, i.e. $\Sigma(x) = \mathbb{I}_N$.

Appendix B: Connection with Information Loss

The concept of *information loss* was introduced in a series of interesting papers [27, 28], as the difference between two mutual informations, $I(x,y) - I(x,z)$, where the random variables y and z are functions of the random variable x. The key point is that both $I(x,y)$ and $I(x,z)$ formally diverge to infinity, but their difference remains finite. Here we show that our renormalized mutual information can be interpreted as an information loss.

Indeed, the diverging mutual information $I(x, y = f(x))$ can be made finite in at least two different ways: either by adding noise to the input variables to obtain $I(x, y = f(x+\varepsilon\lambda))$, or by adding noise to the output variables to get $I(x, z = f(x) + \varepsilon\lambda)$. Here we assume that in both cases λ are Gaussian variables with zero mean and unit variance. A straightforward calculation shows that in the limit $\varepsilon \ll 1$ we have

$$I(x, f(x+\varepsilon\lambda)) = H(y) - KH_\varepsilon - \int dx P_x(x)\ln\sqrt{\nabla f(x)\cdot\nabla f(x)}, \tag{B1}$$

$$I(x, f(x) + \varepsilon\lambda) = H(y) - KH_\varepsilon, \tag{B2}$$

with $H_\varepsilon = \frac{1}{2}\ln 2\pi e\varepsilon^2$.

By subtracting the second equation from the first one and adding $H(y)$, we see that the divergent term KH_ε cancels out, and that we obtain a relation between the our *finite* renormalized mutual information and the information loss:

$$\tilde{I}(x, f(x)) = H(y) + \lim_{\varepsilon\to 0}\left[I(x, f(x+\varepsilon\lambda)) - I(x, f(x)+\varepsilon\lambda)\right]. \tag{B3}$$

According to [27, 28] the limit above represents the information lost by changing the description of x from $f(x+\varepsilon\lambda)$ to $f(x) + \varepsilon\lambda$.

Appendix C: Reparametrization Invariance

In this section, we verify that renormalized mutual information is invariant under feature reparametrization. Consider an invertible function $g(y) : \mathbb{R}^K \to \mathbb{R}^K$and the associated random variable $z = g(y)$. Renormalized mutual information between x and z can be expressed as

$$\tilde{I}(x,z) = H(z) - \int dx P_x(x)\ln\sqrt{|\nabla g(f(x))\cdot\nabla g(f(x))|} \tag{C1}$$

By employing the properties of differential entropy, we can rewrite

$$H(z) = H(y) + \int dx P_x(x) \ln \left| \frac{dg}{dy} \right|.$$

The second term of Eq. (C1) can be expanded via the chain rule of differentiation

$$\nabla g(f(x)) = \frac{dg}{dy} \cdot \nabla f(x)$$

and by using the properties of the determinant

$$|\nabla g(f(x)) \cdot \nabla g(f(x))| = \left| \frac{dg}{dy} \right|^2 |\nabla f(x) \cdot \nabla f(x)|.$$

By putting all together, we get

$$\tilde{I}(x, z) = H(y) - \int dx P_x(x) \ln \sqrt{|\nabla f(x) \cdot \nabla f(x)|} = \tilde{I}(x, y)$$

as we wanted to show.

Appendix D: Feature Extraction in a Low-Dimensional Setting

Here, we show how we implemented Eq. (A1) to extract the feature in Fig. 3 of the main text. In particular, in this very simple case we extract a one-dimensional feature from a two-dimensional space. In a low-dimensional setting, it is still conceivable to discretize the x space in a square region, with lattice constant Δx. We will also discretize y, and let Δy be the size of each bin. We keep $f(x)$ continuous and we parametrize it with a neural network, i.e. $f(x) = f_\theta(x)$, where θ are the weights and the biases of the neurons of the network. To implement Eq. (A1) in TensorFlow, we need to write it as a differentiable function. By looking at Eq. (A1), we see that the first term $H(y)$ must be approximated in some way; the second term can be directly used. The easiest way to approximate $H(y)$ is to estimate $P_y(y)$ with a histogram, and then use it to compute the sum $H(y) = -\Delta y \sum_k P_y(y_k) \log P_y(y_k)$. By definition, we would have

$$P_y(y_k) = \Delta x^2 \sum_{ij} P_x(x_{ij}) \chi_{[y_k - \frac{\Delta y}{2}, y_k + \frac{\Delta y}{2}]}(f(x_{ij}))$$

where $\chi_I(y)$ is the function that is 1 only if $y \in I$ and 0 otherwise. However, χ is not differentiable, and in the regions where it is, it has a zero gradient. This is not convenient for our purpose. So, we replace it with

$$d_k(f(x_{ij})) = \begin{cases} \frac{1}{\Delta y} + \frac{1}{\Delta y^2}(f(x_{ij}) - y_k) & y_{k-1} < f(x_{ij}) \leq y_k, \\ \frac{1}{\Delta y} - \frac{1}{\Delta y^2}(f(x_{ij}) - y_k) & y_k < f(x_{ij}) \leq y_{k+1}, \\ 0 & \text{otherwise.} \end{cases}$$

In other words, instead of assigning each point to a single bin of the histogram, this function assigns it to two bins, in a way linearly proportional to the distance from the center of the bins.

The optimization of Eq. (A1) is performed through gradient ascent: at each step, we calculate $f_\theta(x_{ij})$ for all the points of the x-grid and use it to estimate $P_y(y)$. In particular, we always fix the bounds of the histogram betwen $\min_{ij} f_\theta(x_{ij})$ and $\max_{ij} f_\theta(x_{ij})$. We calculate $\tilde{I}(x, y = f_\theta(x))$ and use backpropagation to update each parameter of the neural network, i.e.

$$\theta' = \theta + \eta \frac{\partial \tilde{I}_\theta}{\partial \theta},$$

where η is a fixed parameters of the algorithm. The procedure is repeated until convergence.

Clearly, the aim of this example was only to show the behavior of our quantity in a very controlled setting. High-dimensional implementations for practical situation will undoubtedly require a more clever technique to approximate the first term of Eq. (A1), as discussed at the end of the main text.