ANNUAL REVIEWS

Annual Review of Control, Robotics, and Autonomous Systems

A Separation Principle for Control in the Age of Deep Learning

Alessandro Achille and Stefano Soatto

Department of Computer Science, University of California, Los Angeles, California 90095, USA; email: achille@cs.ucla.edu, soatto@cs.ucla.edu

Annu. Rev. Control Robot. Auton. Syst. 2018. 1:287–307

First published as a Review in Advance on January 25, 2018

The *Annual Review of Control, Robotics, and Autonomous Systems* is online at control.annualreviews.org

https://doi.org/10.1146/annurev-control-060117-105140

Keywords

deep learning, autonomy, neural network, prediction, representation, invariance, minimality, generalization, learning control, end-to-end training

Abstract

We review the problem of defining and inferring a state for a control system based on complex, high-dimensional, highly uncertain measurement streams, such as videos. Such a state, or representation, should contain all and only the information needed for control and discount nuisance variability in the data. It should also have finite complexity, ideally modulated depending on available resources. This representation is what we want to store in memory in lieu of the data, as it separates the control task from the measurement process. For the trivial case with no dynamics, a representation can be inferred by minimizing the information bottleneck Lagrangian in a function class realized by deep neural networks. The resulting representation has much higher dimension than the data (already in the millions) but is smaller in the sense of information content, retaining only what is needed for the task. This process also yields representations that are invariant to nuisance factors and have maximally independent components. We extend these ideas to the dynamic case, where the representation is the posterior density of the task variable given the measurements up to the current time, which is in general much simpler than the prediction density maintained by the classical Bayesian filter. Again, this can be finitely parameterized using a deep neural network, and some applications are already beginning to emerge. No explicit assumption of Markovianity is needed; instead, complexity trades off approximation of an optimal representation, including the degree of Markovianity.

1. INTRODUCTION

Say you have a time series of data and wish to store a function of it that has constant complexity, which we call a representation, that is useful for a prediction or control task. Here, useful means that the representation retains the same amount of information about the task as the original data.[1]

For example, if y_t is the output of a linear time-invariant system with true finite-dimensional state x_t, driven by white, zero-mean Gaussian noise, then a sufficient representation $\hat{x}_t$ of the data $y^t := \{y_0, \dots, y_t\}$ for the task z_t—for instance, prediction $z_t = y_{t+1}$—is the mean and covariance of the posterior density $p(x_t|y^t)$ (1) or, equivalently, the posterior itself (we further discuss this equivalence below). This representation summarizes all past history of the data for the purpose of the task (in this case, predicting its future).[2] In other words, given the representation, past data are independent of future data. Such independence makes it possible to separate the inference of the state given the measurements from the control design given the estimated state (5).

Such a separation principle has served the practitioner well over the years but has left us with few tools for cases when the underlying assumptions are not satisfied: What if noise is not additive or Gaussian? What if the true state is high dimensional or even infinite dimensional? What if the data are also high dimensional, and almost all of the data are irrelevant for the control task?

Unfortunately, these are conditions that the practitioner of robotics and autonomous systems faces routinely: The task may include navigation in an unknown environment populated by objects whose shape is described by (infinite-dimensional) surfaces and reflectance functions; the data may include images with millions of channels (pixels), predicting most of which is irrelevant to the navigation task; and nuisance factors may include occlusion and changes of illumination and pose, which are far from white, zero-mean Gaussian noise. Does a separation principle exist for this kind of scenario? Is it still possible to infer a bounded-complexity function of the data that can be stored in memory in lieu of the data, with no information loss?

To be sure, there have been many attempts to answer these questions. In sufficient dimensionality reduction (6, 7), one aims to identify small-dimensional statistics (e.g., projections) that summarize the data. Similarly, the classical use of invariants in image analysis is to remove redundancy from the data by mapping it to the quotient space, which for images can be a thin set (8). These approaches have had limited impact in robotics and autonomous systems. Sufficient reductions are either too restrictive (linear projections) or too hard to compute and difficult to use. But what if we go the opposite way of dimensionality reduction? What if we instead increase the dimensionality beyond that of the data, already in the millions? What if, instead of computing statistics (deterministic functions of the data), we allow representations to be arbitrary stochastic functions?

At least for simpler tasks, such as classification (9) or control in a finite setting, deep neural networks (DNNs) with hundreds of millions of parameters have shown remarkable empirical success. Can we leverage this success to infer representations of time series specifically for filtering and control tasks? Is there a theoretical framework that explains why large networks would work well for control?

Using a large network seems ill advised at first: The bias-variance dilemma (10) states that as we increase the complexity of a model inferred from finitely sampled data, the model's ability

[1] For such a representation to exist, we must assume that the data satisfy a Markov model, an assumption to which we return below.

[2] If the model is not known, then the representation includes a constant component that belongs to an equivalence class of realizations (2), and there is an elegant geometry that is exploited in subspace system identification (3). Even if there is no true finite-dimensional state, under the Markovian assumption with Gaussian inputs of known dimension, one can infer a finite-dimensional predictor along with the state and model parameters (4).

to capture the underlying distribution degrades, a phenomenon referred to as overfitting, which seems at odds with the empirical success of DNNs (11). However, if the network complexity is measured by information content, rather than dimension, a well-trained DNN for classification faithfully obeys the bias-variance trade-off (12), and relaxing representations to be stochastic functions has the double advantage of simplifying the computation of information quantities and analyzing the properties of the resulting representations.

In this article, we study representations for robotics and autonomous systems using tools from statistics and information theory and using DNNs as the class of functions implementing them (realizations). We first review the simple case of a model with trivial dynamics, to introduce the machinery of DNNs, and then extend it to the dynamic case.

1.1. Outline of the Review

In Section 2, we introduce the defining properties of representations and formalize the notions of sufficiency, minimality, invariance, and separation. Since representations that are minimal and sufficient do not exist in general in finite dimensions (13), we start from the posterior, which is minimal sufficient (14) but infinite dimensional, and frame the problem of learning representations as an approximation problem where complexity is modulated explicitly in the information bottleneck Lagrangian (IBL). This is a cost functional to be minimized with respect to a class of functions in a sufficiently rich set.

Like many other function classes, DNNs are universal approximants in the limit where the number of parameters goes to infinity. However, they enjoy a peculiar coupling between the model parameters and the properties of the learned representation that make them better than most when we want the representation to be invariant to nuisance variability and to have components that are maximally independent (disentanglement). Section 4 provides a succinct introduction to DNNs, to the extent needed to follow the rest of the article.

Section 3 presents a series of core results that explain in what sense deep networks are approximations of optimal representations for static systems (15). Specifically, through the use of the IBL, we formalize the trade-off between the complexity of the data representation and the error we commit when we use this representation to solve the task in lieu of the original data.

However, at first sight the IBL does not address two important properties of a representation, invariance and disentanglement, which we should then deal with separately. Instead, we show that, given sufficiency, invariance is equivalent to minimality. We also show that the IBL is equivalent to the cross-entropy loss typically used for classification tasks in deep learning except for an added regularizer, thus creating an important link between the (information-theoretic) optimal representations and the deep learning practice. Furthermore, some heuristic methods used in optimizing deep networks [stochastic gradient descent (SGD), dropout, and variants of dropout] approximate this regularizer (12). We then show that, somewhat counterintuitively, stacking layers of neural networks increases the minimality of the representation and therefore its invariance. This is tied to the architecture of deep networks and partly explains their success.

Architecture design is also critical in coupling the optimization process, where the IBL is minimized with respect to the parameters of the network (weights), with the desirable properties of the resulting representations (activations), which we outline in Section 2. Specifically, in Section 5, we describe an inequality that links the activations of a deep network (a representation of the test datum) and its weights (a representation of the training set). This duality also sheds light on the generalization properties of DNNs.

Section 6 extends the model to dynamical systems. By explicitly introducing a task variable, which is in general separate from the data, we open the possibility of drastically more efficient

representations than those sufficient for future data prediction, while still learning end to end with a simple filter.

In Section 8, we discuss some properties and limitations of the representation proposed. Specifically, we discuss the limitations of the Markovianity assumptions in classical models and how the proposed model partially overcomes them by trading them off against complexity costs.

1.2. Related Work

Deep learning is impacting many areas of engineering and science, including time-series forecasting (16); it shows promise especially when some of the most common hypotheses underlying conventional methods, such as Markovianity, are not satisfied (17). Several studies have examined extensions of classical Bayesian filtering, including using neural networks, but the resulting approaches had drawbacks. First, the complexity of the update rule, which in the classical Bayesian setting requires computing the posterior of the data given the hidden state, is problematic for high-dimensional data types such as images. Second, the only task considered is prediction of the data, which is usually overkill when the actual task is, say, control: One does not need to model the complex reflectance properties of the world or predict the color of each pixel to decide whether to steer a vehicle to the right. Finally, existing methods do not allow an explicit trade-off between the complexity of the hidden state and the quality of the prediction error. Notice, however, that variational Bayesian methods can be used as a partial solution to the first problem (18, 19), as long as the function class covers the underlying data distribution, which remains an open problem for natural images.

Among other methods, the deep Kalman filter (18) assumes the existence of a Gaussian latent state and nonlinear transformations that explain the observations and use a variational autoencoder to infer such a model. This choice is restrictive, as the only task allowed is the reconstruction of the measurements. Also, the method focuses on batch system identification, whereas we are interested in a causal, online scheme.

In a study more directly related to our approach, Langford et al. (20) suggested that, rather than finding a (generally complex) hidden state, we could focus on finding a statistic x_t of the past data y^t that separates past data from future predictions. Such a statistic can be learned and updated without using Bayes's rule, thereby avoiding the complex computations of the data posterior $p(y_t|x_t)$. However, their analysis is restricted to linear models, and the task is restricted to the prediction of future data.

Constructing deterministic functions of the data (statistics) that separate the past from the future requires $N+N^2$ (embedding) dimensions [the mean and covariance maintained by a Kalman filter (21), where N is the dimension of the state space]; however, as we show below, a stochastic representation can make do with N dimensions, at the cost of maintaining samples from the distribution, as in a particle filter. Sigma-point filters (22) are deterministic sample-based representations that fall between these two cases. In this case as well, the task is prediction of the measurements. We allow the task to be more general, including the case where one does not care about being able to reproduce every channel of the measurements (e.g., the color of every pixel in the image) but instead cares only about a small projection or quotient of the data with respect to the action of nuisances. Our model also allows even more flexibility relative to the strong assumption of Markovianity implicit in the classical filtering equations. The inferred state can be thought of as a separator but only for the measurements, as opposed to a more general task, which makes the problem less tractable than in our case, because the statistics that matter for control typically have far less complexity than the data. Also, the proofs provided apply only to minimal realizations. In our model, we trade off Markovianity and complexity, which is not contemplated in the classical filtering equations.

The trade-off we seek between the complexity of the representation and sufficiency for future prediction is also closely related to the minimum-information linear-quadratic Gaussian (LQG) control of Fox & Tishby (23), which explicitly accounts for the agent having a representation of bounded complexity. They address only the LQG case but give a complete account of it. Similarly, Tiomkin et al. (24) and Rubin et al. (25) deal with capacity costs but in continuous time. The general principles were laid out by Fox et al. (26).

The theory we describe here emphasizes that, in order to obtain efficient representations of the data, we should focus on a specific task, such as control, rather than predicting high-dimensional future data. Following a related idea, Dosovitskiy & Koltun (27) assumed that there is a low-dimensional vector of measurements separate from the actual measured data that can be easily obtained and on which the control loss depends linearly. In our parlance, predicting the future is a task sufficient for control and therefore allows us to learn a sufficient representation for control. In particular, optimal control reduces to minimum-prediction error, and one can simply train a network to predict future measurements conditioned on the current policy and given what control action is going to be taken at the current time. Using this technique for control shows state-of-the-art performance on video games.

Finally, we study the quantity of information that the observed data contain about the parameters of the system plays. This quantity was considered by Houthooft et al. (28), who used a variational approximation similar to ours to measure the information content in the data using a neural network, which then learned a control policy for exploration that maximized this information quantity. In this case, the model is a constant parameter, akin to an assumption of time invariance and equivalent to Markovianity.

1.3. Preliminaries

We denote the history of a process from time t_0 to t by $y_{t_0}^t = \{y_{t_0+1}, \dots, y_t\}$, where we omit the subscript when $t_0 = 0$. Thus, y^t denotes the measured data up to time t, while z_t denotes the quantity of interest (task) at that time, which could be the value of the measurements at a future time, $z_t = y_{t+\tau}$. We consider trivial dynamics at first, with each $(y_t, z_t) \sim p_\theta(y, z)$ sampled independently and identically distributed (i.i.d.) from an unknown density p_θ.

For random variables y, z, x, we denote the expectation of y with respect to the measure $p(y)$ by $\mathbb{E}_p[y]$, the (differential) Shannon's entropy by $H(y) = \mathbb{E}_p[-\log p(y)]$, conditional entropy by $H(y|z) := \mathbb{E}_y\mathbb{E}_x[-\log p(x|y)] = H(y, z) - H(z)$, (conditional) mutual information by $I(y; z|x) = H(y|x) - H(y|z, x)$, Kullback–Leibler (KL) divergence by $\mathrm{KL}(p(y) \,\|\, q(y)) = \mathbb{E}_{y\sim p(y)}[\log p(y)/q(y)]$, and cross-entropy by $H_{p,q}(y) = \mathbb{E}_{y\sim p(y)}[-\log q(y)]$. The total correlation of x, denoted by $\mathrm{TC}(x)$, is defined as

$$\mathrm{TC}(x) := \mathrm{KL}(p(x) \,\|\, \textstyle\prod_i p(x_i)),$$

where $p(x_i)$ are the marginal distributions of the components of x. Notice that $\mathrm{TC}(x)$, also known as multi-information or multivariate mutual information, is zero if and only if the components of x are independent, in which case we say that z is disentangled. We make use of the following identity:

$$I(x; y) = \mathbb{E}_{y\sim p(y)}\mathrm{KL}(p(x|y) \,\|\, p(x)).$$

We say that y, x, z form a Markov chain, indicated with $z \to x \to y$, if $p(z|y, x) = p(z|x)$. The data-processing inequality for a Markov chain $y \to x \to z$ ensures that $I(y; x) \geq I(y; z)$.

We define a nuisance to be any random variable that affects the observed data y but is not related to the task, $z \perp\!\!\!\perp n$, or, equivalently, $I(z; n) = 0$. Similarly, we say that the representation

x is invariant to the nuisance n if $x \perp\!\!\!\perp n$, or $I(x;n) = 0$. When x is not strictly invariant but minimizes $I(x;n)$ among all sufficient representations, we say that it is maximally insensitive to n.

For instance, to recognize the same object in two views, both vantage point and illumination are nuisances: They affect the data but do not inform the task variable z. On the other hand, to navigate, vantage point is the variable of interest, whereas the visual appearance of surrounding objects is a nuisance.

The data can always be written as a function of the task and of all nuisances affecting it (12). Specifically, given a joint distribution $p(y,z)$, where z is a discrete random variable, we can always find a random variable n independent of z such that $y = f(z,n)$ for some deterministic function f.

Given random variables y, z, and x with joint density $p(y,z,x)$, we say that x is sufficient of y for the task z if we have the Markov chain $y \to x \to z$, i.e., if $p(z|y,x) = p(z|y)$. We say instead that the posterior of x, $p(x|y)$, is sufficient of y for z if $p(z|y) = \int p(z|x)p(x|y)\mathrm{d}x$. Notice that the posterior of a sufficient representation is in turn always sufficient, since $p(z|y) = \int p(z,x|y)\mathrm{d}x = \int p(z|x,y)p(x|y)\mathrm{d}x = \int p(z|x)p(x|y)\mathrm{d}x$. The converse does not hold in general but holds in the important case in which x is a deterministic function of y.

For example, in visual correspondence, the (test) datum y_t is an image (or a part of an image, such as a patch or bounding box around a feature point), while the task $z \in \{0,1\}$ is a binary decision as to whether the image portrays the same scene as a previously seen (training) image $y_{t-\tau}$ for some $\tau > 0$. Under the admittedly restrictive assumptions of corresponding images being related by a similarity transformation of the domain (translation, rotation, and scaling of the image) and a similarity transformation of its range (contrast transformations), a sufficient invariant representation can be formally computed in closed form and approximated numerically, as done by Dong & Soatto (29). In particular, the domain-size pooling–scale-invariant feature transform (DSP-SIFT) consists of a histogram of gradient orientations, which are invariant to contrast, with translation and scale locally marginalized (averaged) and orientation selected according to a data-dependent criterion (direction of maximum average gradient norm). In this case, vantage point, modeled simplistically as a similarity transformation of the image domain, is a nuisance to which the representation is invariant.

On the other hand, in vision-based navigation or simultaneous localization and mapping (SLAM), the vantage point $z_t \in \mathrm{SE}(3)$, i.e., the position and orientation of the sensor platform relative to an inertial reference frame, *is* the task. The data $y^t = \{y_0, \ldots, y_t\}$ represent the history of images measured up to the current time. While the posterior density $p(z_t|y^t)$ appears sufficient for z_t, since it contains all information we have about z_t up to the present, as we show below, it is not sufficient to update the representation given future measurements. Rather, a sufficient representation must include the approximate speed and position of the sensor and the position of observed feature points in the global reference frame. Consequently, most SLAM systems maintain an approximation of the joint posterior of a sparse attributed point cloud and the sensor pose—for instance, using an extended Kalman filter or a particle filter. The attribute of each point—for instance, a DSP-SIFT descriptor—must be sufficient to establish correspondence, and the positions of the points serve as a reference for camera pose. The number of points in the representation must be sufficient to define a reference frame visible with high probability at any future time.

In this article, we focus on the general case of sufficient posteriors and abuse the notation to refer to both the random variable x and the posterior $p(x|y)$ as being sufficient.[3]

[3] An equivalent characterization using conditional expectations is to say that x is sufficient of y for z if $\mathbb{E}[f(z)|y,x] = \mathbb{E}[f(z)|x]$ for any measurable function f and, similarly, that the posterior is sufficient if $\mathbb{E}[\mathbb{E}[f(z)|x]|y] = \mathbb{E}[f(z)|y]$ for any f.

2. DESIDERATA FOR REPRESENTATIONS

We call a representation x of the data y any stochastic function $x \sim p(x|y)$ of y. Ideally, we would like x to be sufficient for the task z; that is, all the information that y contains about the task should also be contained in x, or $I(x;z) = I(y;z)$. To avoid squandering resources, the representation x should also be minimal; that is, $I(y;x)$ should be smallest among all sufficient representations x. Note that we are defining "small" in terms of the information content, not the dimension, of x. Moreover, we would like x to be invariant to a nuisance n, $I(x;n) = 0$, or, if that is not possible, at least maximally insensitive to it—i.e., $I(x;n)$ is minimized. Note that we require invariants to be uninformative, not constant, with respect to nuisance variability. We impose no requirement on identifiability and harbor no hope of uniqueness of representations. However, to facilitate their use, we do wish for the components of x to be maximally disentangled; that is, we want $\mathrm{TC}(x) = \mathrm{KL}(p(x) \,\|\, \prod_i p(x_i))$ to be minimized.

The first two properties are satisfied by any minimal sufficient representation, which can be found by solving

$$\begin{aligned} \min_{p(x|y)} \quad & I(y;x) \\ \text{subject to} \quad & H(z|x) = H(z|y) \end{aligned}$$

with respect to the class of posteriors $p(x|y)$, or minimizing the corresponding IBL (30):

$$\mathcal{L} = \underbrace{H(z|x)}_{\text{cross-entropy}} + \beta \underbrace{I(x;y)}_{\text{regularizer}}. \qquad 1.$$

The IBL trades off sufficiency and minimality, regulated by β, and can be optimized efficiently when the x is parameterized by a neural network (15, 31). However, we are also interested in the two other properties, invariance and disentanglement, that are not explicit in the IBL and are the focus of the next section.

3. LEARNING INVARIANT AND DISENTANGLED REPRESENTATIONS

The following key result from Achille & Soatto (12) connects the minimality of a representation to its invariance to nuisances.

Proposition 1. Let n be a nuisance affecting the data y. Then, for any representation x of y, we have

$$\underbrace{I(x;n)}_{\text{invariance}} \le \underbrace{I(x;y)}_{\text{minimality}} - \underbrace{I(y;z)}_{\text{constant}},$$

where the right-hand side is minimized when x is minimal. Moreover, there always exists a particular nuisance n such that equality holds up to a (generally small) residual ϵ, that is,

$$I(x;n) = I(x;y) - I(y;z) - \epsilon,$$

where $\epsilon := I(x;z|n) - I(y;z)$. In particular, $0 \le \epsilon \le H(z|y)$,[4] and $\epsilon = 0$ whenever x is a deterministic function of y. Under these conditions, a sufficient statistic x is invariant (maximally insensitive) to nuisances if and only if it is minimal.

[4]Notice that since $\epsilon \le H(z|y)$, and usually $H(z|y) \ll I(y;x)$, we can generally ignore the extra term.

This result implies that, rather than manually imposing invariance to nuisances in the representation, which is usually difficult, we can construct invariants by simply reducing the amount of information that x contains about y, while retaining sufficiency. As discussed above, this can be done by using a neural network to minimize the IBL (15).

We analyze deep networks in Section 4, but this result, together with the data-processing inequality, already suggests an advantage in stacking multiple intermediate representations into layers. In fact, suppose that we have a Markov chain of representations

$$y \to x_1 \to x_2$$

such that there is an information bottleneck between x_2 and x_1, that is, $I(x_1; x_2) < I(x_1; y)$. Then, if x_2 is still sufficient, it is necessarily more minimal, and therefore more invariant to nuisances, than x_1. Notice, moreover, that bottlenecks are easy to create, either by reducing the dimension so that $\dim(x_2) < \dim(x_1)$ or by introducing noise between x_2 and x_1. This is indeed common practice in designing and training deep networks, which concatenate multiple layers

$$y \to x_1 \to x_2 \to \cdots \to x_L$$

so that, whenever layer x_L is sufficient of y for z (which is imposed by the training loss), then x_L is more insensitive to nuisances than all the preceding layers.

This also relates to the notion of actionable information (32), which is the entropy $\mathcal{H}(y) := H(f(y))$ of a deterministic maximal invariant $f(y)$ of the observed data. In the special case when the representation $x = f(y)$ is deterministic, a representation that minimizes the IBL also maximizes actionable information (12). Interestingly, the IBL can also be modified to favor deterministic representations of the data by replacing the mutual information cost $I(y; x)$ with the entropy $H(x)$ of the representation (33).

Finally, Achille & Soatto (15) showed that, if the mutual information $I(y; x) = \mathbb{E}_y \mathrm{KL}(q(x|y) \,\|\, xq(x))$ is naively approximated by substituting the unknown joint marginal $q(x)$ with a factorized prior $\tilde{q}(x) = \prod_i q(x_i)$, then the modified IBL not only is easier to compute, but also can be used to bound the total correlation of the representation x. Therefore, minimizing the simplified IBL yields a representation that trades off sufficiency with complexity, invariance, and disentanglement. This is particularly interesting because it implies that simply reducing information when using a factorized prior yields not only invariance but also disentanglement, even without explicitly adding a term in the loss function (IBL).

Up to this point, representations have been generic (infinite-dimensional) functions of past data. In the next section, we introduce the basic elements of DNNs, the class of functions we choose to approximate representations that minimize the IBL.

4. LEARNING WITH DEEP NEURAL NETWORKS

In this section, we sketch the very basics of deep learning, by describing first the class of functions realized by DNNs and then the choice of functionals and optimization schemes used to determine their parameters. In Section 5, we then show how this process, despite being agnostic about desirable properties of the representations outlined in the previous sections, manages to achieve just that by exploiting a peculiar information duality between the weights and the activations of the network.

4.1. The Function Class of Deep Neural Networks

A DNN is a parameterized class of nonlinear functions obtained by composing multiple layers. Each layer implements a linear transformation of its input, which is the output of the previous

layer, followed by a (generally element-wise) nonlinearity. Specifically, let $y := x^0 \in \mathbb{R}^d$ be the input data, and let $W^k \in \mathbb{R}^{d_{i-1} \times d_i}$ be a matrix, where $d_0 := d$. Then, we define the activations (output) of the kth layer as $x^k = \phi_k(W^k x^{k-1})$, where ϕ_k is a nonlinear function. A common choice for the nonlinearity is $\phi_k(x) = \max(0, x)$, also called a rectified linear unit (ReLU). The output x^K of a network with K layers is the function

$$F(y; w) = \phi_K(W^K \phi_{K-1}(W^{K-1} \dots \phi_1(W^1 y) \dots)),$$

where $w = \{W^1, \dots, W^k\}$ is the set of parameters, or weights, of the network. Each x^k can be considered a representation of the original input y, and its components are generally called features (or feature maps, activations, or responses). By the data-processing inequality, x^k contains no more information than y; however, as shown below, in a well-trained network we expect x^k to contain all the information that the data contain about the task. Since the output of the network is often a (conditional) probability distribution [e.g., the probability $p(z|y)$ of a label z given the image y], the last nonlinearity is usually a softmax nonlinearity, $\text{softmax}(x)_i = e^{y_i}/(\sum_j e^{y^j})$, which ensures that the output of the network is positive and sums to one.

When the input y has some particular structure, such as an image, the linear transformation W^k can be chosen to exploit this structure. For example, when y is an image, it is common to choose W^k to be a set of convolutions. Networks using convolutional maps, known as convolutional neural networks (CNNs), have the notable property that their features are invariant to translations (34) and have considerably fewer parameters (the number of parameters depends only on the sizes of the filters, which are generally small, rather than on the size of the image). Aside from reducing the size of the parameter space, the use of convolutions has a drastic, and not yet fully understood, effect in achieving desirable properties of the networks when operating on imaging data (35).

4.2. The Loss Function and Optimization

The output of a network is usually interpreted as a probability distribution $q(z|y, w)$ over the inference target z (e.g., the label of an image or the position of an object). Per Soatto & Chiuso (35), if that output approached the true posterior, it would be a minimal sufficient representation.

When z is a discrete random variable, this identification can be done directly by letting the output $F(y, w)$ of the network be a probability vector (or an unnormalized likelihood function). When z is continuous, we can choose a family of parameterized distributions and let the network output the parameters (e.g., mean and variance for a normal distribution). In both cases, we think of a deep network as a map $y \mapsto F_w(y) := q(\cdot|y, w)$ where, absent any system dynamics, the parameters w are constant and usually determined by maximizing the log-likelihood of the observed data, which leads to the cross-entropy loss

$$L(w) = H_{p,q}(z|y, w) = \frac{1}{t} \sum_{i=1}^{t} -\log q(z_i|y_i, w).$$

Notice that the cross-entropy loss can be decomposed as

$$H_{p,q}(z|y, w) = H_p(z|y) + \text{KL}(p(z|y) \,\|\, q(z|y, w)).$$

Since all terms are positive and only the KL divergence depends on w, we can conclude that $L(w)$ is minimized if and only if $q(z_i|y_i, w) = p(z_i|y_i)$ on all observed samples, giving an alternative justification for the use of this loss.

Minimizing the loss $L(w)$, and hence determining the weights w, is usually done using SGD. We start by randomly initializing the parameters w (36). Then, at each step k, a random subset

(mini-batch) $(y_{i_k}^{i_k+b}, z_{i_k}^{i_k+b})$ of size b, with $i_k \sim \text{unif}(0, t-b)$, is sampled from the observed data (y^t, z^t), and we compute the gradient g^k relative to the mini-batch:

$$g^k = \frac{1}{b}\nabla_w H_{p,q}(z_{i_k}^{i_k+b} | y_{i_k}^{i_k+b}, w) = \frac{1}{b}\sum_{j=0}^{b} -\nabla_w \log q(z_{i_k+j} | y_{i_k+j}, w).$$

Since $\nabla_w L(w) = \mathbb{E}[g^k]$, we can see g^k as an unbiased (but high-variance, or "noisy") estimate of the real gradient of the original loss function with respect to w. This can be computed efficiently since it requires computing the gradients on only b samples rather than the whole collection of observed data, which can number in the millions. The weights are now updated using $w \leftarrow w + \eta_k g^k$, where $\eta_k > 0$ is called the learning rate. When the loss function is strongly convex, the gradients are Lipschitz, and the learning rate decreases as $\eta_k = 1/k$, SGD converges to the global optimum of the loss with convergence rate $O(1/t)$ (37).

There are two main challenges in carrying out this optimization: (*a*) The loss function is highly nonconvex, and therefore SGD can converge (through annealing of the learning rate) to suboptimal solutions, and (*b*) even if a global minimum is found (the training loss is zero), the parameters could be overfitting the data, meaning that while w minimizes the loss on the observed data, the loss evaluated on unseen (future) data could be much larger.

The first problem is partly addressed by SGD itself: Because of the noise added in the computation of the gradient by SGD, the optimization typically settles on extrema that are close to the global minimum in value. Variants of SGD include using Nesterov's momentum (37), which generally yields faster training and improved performance of the network. Other algorithms, like RMSProp and Adam (38), use the gradient history to reduce the variance in the estimate of the gradient, which is also adapted to the local geometry of the loss function. While in some tasks, such as stochastic optimal control (reinforcement learning) (39), these algorithms show drastically improved performance (as expected), on image classification and similar tasks, the simpler SGD with momentum can still outperform them, suggesting that the noise added by SGD plays an important positive role in the optimization. There is at present a considerable amount of activity but a dearth of results in characterizing the topological and geometric properties of the loss function and designing algorithms that can exploit it to converge to minima that yield good generalization performance, as we discuss in Section 5.1. Generalization, or lack thereof (overfitting), is the second problem, which we discuss in more detail in the next section.

5. DUALITY AND GENERALIZATION

One of the main problems in optimizing a DNN is that the cross-entropy loss in notoriously prone to overfitting: The loss is small for (past) training data (thus, optimization is successful) but large on (future) test data, indicating that the training process has converged to a function that is far from being an optimal representation.

We can gain insight about the possible causes of this phenomenon by looking at the following decomposition of the cross-entropy (12):

$$\underbrace{H_{p,q}(z^t|y^t, w)}_{\text{cross-entropy}} = \underbrace{H_p(z^t|y^t, \theta)}_{\text{intrinsic error}} + \underbrace{I(\theta; z^t|y^t, w)}_{\text{weights sufficiency}} + \underbrace{\mathbb{E}_{w,y^t}\text{KL}(q(z^t|y^t, w) \,\|\, p(z^t|y^t, w))}_{\text{efficiency of model}} - \underbrace{I(z^t; w|y^t, \theta)}_{\text{memorization/overfitting}}, \quad 2.$$

where $w \sim q(w|y^t, z^t)$. The first term on the right-hand side of Equation 2 relates to the intrinsic error and depends only on p_θ, the second term measures how much of the information that past

data contain about the parameter θ is captured by the weights, and the third term relates to the efficiency of the model and the class of functions f_w with respect to which the loss is optimized. The last (and only negative) term relates to how much information about the labels is memorized in the weights, regardless of the underlying data distribution. Absent any intervention, the left-hand side of Equation 2 can be minimized by just maximizing the last term, i.e., by memorizing the data set, which amounts to overfitting and yields poor generalization. Traditional machine learning practice suggests that this problem can be avoided by reducing the complexity of the model or by regularizing its parameters. On the other hand, modern architectures, even when using standard regularizers, are still prone to memorizing the training labels.

Memorization can, however, be prevented by adding the last term back to the loss function, leading to a regularized loss $H_{p,q}(z|y,w) + I(z;w|y,\theta)$, where the negative term on the right-hand side is canceled. However, computing, or even approximating, the value of $I(z,w|y,\theta)$ is at least as difficult as fitting the model itself.

To overcome this problem, consider $\mathcal{D} = (y^t, z^t)$, the collection of all past data that we are using to infer the model parameters w. Notice that to successfully learn the distribution p_θ, we only need to memorize in w the information about the latent parameters θ; that is, we need $I(\mathcal{D};w) = I(\mathcal{D};\theta) \leq H(\theta)$, which is bounded above by a constant. On the other hand, to overfit, the term $I(z;w|y) \leq I(\mathcal{D};w|\theta)$ needs to grow linearly with the number of training samples N. We can exploit this fact to prevent overfitting by adding a Lagrange multiplier β to make the amount of information constant with respect to N, leading to the regularized loss function

$$\mathcal{L}(p(w|\mathcal{D})) = H_{p,q}(z|y,w) + \beta I(w;\mathcal{D}), \qquad 3.$$

which is, remarkably, the same IBL in Equation 1 but now interpreted as a function of w rather than x. Under appropriate assumptions on the form of the posterior $q(w|\mathcal{D})$, the term $I(w;\mathcal{D})$ can be computed in closed form, and we can optimize Equation 3 efficiently (12, 40).

Thus, as we have seen, the IBL emerges as a natural criterion both for inferring a representation of the test datum y that is sufficient and invariant (with no explicit notion of overfitting) and for inferring a representation w of the training data set (past data) $\mathcal{D}$ that avoids overfitting (with no explicit notion of invariance). A natural question, which we address in Section 5.2, is whether and (if so) how these two representations and their corresponding IBLs are related to each other.

Note that the IBL above is not the one usually described in the information bottleneck literature, including various attempts to develop a theory of deep learning. Above, we considered both the weights and the data set as random variables, and the information bottleneck relates to the weights as a representation of the data set, rather than relating the activations as a representation of the test datum, as is customary in the information bottleneck literature. This seemingly subtle but radical departure from current practice was first introduced by Achille & Soatto (12) and is key to relating generalization to representation learning and to arriving at a consistent theory of representation learning. In the next remark, we arrive at similar conclusions using entirely different tools.

Remark 1 (alternative derivation of the information bottleneck Lagrangian for the weights). An alternative approach to the generalization problem is to use the probably approximately correct (PAC)–Bayes framework (41) to bound and minimize the future testing error, rather than directly minimizing the training error. This can be achieved by finding a posterior distribution $q(w|\mathcal{D})$ over the weights that is closed to some fixed prior $p(w)$ while still being able to fit the training data (41). Concretely, this reduces to minimizing the loss function

$$L(q(w|\mathcal{D})) = H(z|y,w) + \beta\, \mathbb{E}_{\mathcal{D}} \mathrm{KL}(q(w|\mathcal{D}) \,\|\, p(w)). \qquad 4.$$

The sharpest PAC–Bayes upper bound on the test error is obtained when the prior $p(w)$ is chosen to be exactly the marginal distribution $q(w) = \mathbb{E}_{\mathcal{D}} q(w|\mathcal{D})$ of the weights over all data sets $\mathcal{D}$ and trainings of the network (12, 41). In this case, using the identity $I(w;\mathcal{D}) = \mathbb{E}_{\mathcal{D}}\mathrm{KL}(q(w|\mathcal{D}) \,\|\, q(w))$, Equation 4 reduces precisely to the IBL in Equation 3 (11).

Equation 4 also admits another important special case: For $\beta = 1$, it reduces to the evidence lower bound (ELBO)

$$-\log p(z^t|y^t) \le H_{p,q}(z^t|y^t, w) + \mathrm{KL}(q(w|\mathcal{D}) \,\|\, p(w)),$$

which can be used to find an approximation $q(w|\mathcal{D})$ of the real Bayesian posterior $p(w|\mathcal{D})$ of the network weights given the data set $\mathcal{D}$ and the prior $p(w)$ (40). The use of the PAC–Bayes framework to study generalization properties of deep networks and their connection to the geometry of the loss function, which we consider in the next section, has also been championed by Dziugaite & Roy (42), albeit without the connection to the IBL. They further used the theory to derive strong bounds on the test error of the network.

5.1. Information, Generalization, and Flat Minima

Thus far, we have suggested that adding the explicit information regularizer $I(w;\mathcal{D})$ prevents the network from memorizing the data set and thus avoids overfitting, which was also confirmed empirically by Achille & Soatto (12). However, common networks are not typically trained with this information regularizer, thus seemingly undermining the theory. However, even when not explicitly controlled, $I(w;\mathcal{D})$ is implicitly regularized by the use of SGD (43). In particular, empirical evidence suggests that SGD biases the optimization toward flat minima—local minima whose Hessian has mostly small eigenvalues. These minima can be interpreted exactly as having low information $I(w;\mathcal{D})$, as suggested early on by Hochreiter & Schmidhuber (44). As a consequence of previous claims, flat minima can be seen as having better generalization properties.

More precisely, let $\hat{w}$ be a local minimum of the cross-entropy loss $H_{p,q}(z|y, w)$ and let $\mathcal{H}$ be the Hessian at that point. Then, under suitable assumptions on the form of the posterior, for the optimal choice of the posterior parameters we have (12)

$$I(w;\mathcal{D}) \le \frac{1}{2} K[\log \|\hat{w}\|_2^2 + \log \|\mathcal{H}\|_* - K \log(K^2\beta/2)],$$

where $K = \dim(w)$ and $\|\mathcal{H}\|_* = \mathrm{tr}(\mathcal{H})$ denotes the nuclear norm of the matrix. Therefore, the information in the weights is upper bounded by the nuclear norm (and hence the flatness) of the Hessian. Notice that a converse inequality—that is, low information implies flatness—need not hold, so sharp minima can in principle generalize as well.

In the next section, we show that the quantity of information on the weights is connected not only to the geometry of the loss function but also to the minimality (invariance) and disentanglement of the activations. In particular, this shows that weight regularization, whether implicit (SGD) or explicit (IBL), biases the optimization toward good representations.

5.2. The Duality of the Representations

The core link between information in the weights, and hence the flatness of the local minima, minimality of the representation, and disentanglement, can be described by the following proposition from Achille & Soatto (12) for the case of a single layer.

Proposition 2. Let $x = Wy$ be a single layer of a network. Under opportune hypotheses on the form of $q(W|\mathcal{D})$, we can find a strictly increasing function $g(\alpha)$ such that we have

the uniform bound

$$g(\alpha) \leq \frac{I(y;x) + \mathrm{TC}(x)}{\dim(x)} \leq g(\alpha) + c,$$

where $c = O(1/\dim(y)) \leq 1$ and α is related to $I(w;\mathcal{D})$ by $\alpha = \exp\{-I(W;\mathcal{D})/\dim(W)\}$. In particular, $I(y;x) + \mathrm{TC}(x)$ is tightly bounded by $I(W;\mathcal{D})$ and increases strictly with it.

Using the Markov property of the layers, we can now easily extend this bound to multiple layers. Let W^k for $k = 1, \ldots, L$ be weight matrices, and let $x_{i+1} = \phi(W^k x_k)$, where $x_0 = y$ and ϕ is any nonlinearity. Under the same assumptions as the previous result, one can prove

$$I(x_L;y) \leq \min_{k<L} \left\{ \dim(x_k) \Big[g(\alpha^k) + 1 \Big] \right\},$$

where $\alpha^k = \exp\{-I(W^k;\mathcal{D})/\dim(W^k)\}$.

Together with the results of Section 3, this implies that regularized networks containing low information in the weights automatically learn a representation of the input that is both more invariant to nuisances and more disentangled. Moreover, by Section 5.1, SGD is biased toward such representations.

This result is important because it establishes connections between the weights, which are a representation of past data, given and used to optimize a loss function that knows nothing about sufficiency, minimality, invariance, or disentanglement, and representations of future data, which emerge to have precisely those properties. Such connections are peculiar to the class of functions implemented by DNNs and do not apply to any generic function class.

Finally, we have all the elements to extend the notion of representation and the optimization involved in inferring it (which encompasses system identification and filtering) to a dynamic setting.

6. REPRESENTING TIME SERIES

In this section, we consider the case where the data are not drawn i.i.d. from a distribution with constant underlying parameters. Instead, we assume that the representation can evolve over time according to a probability law that does not.

6.1. A Hidden-State Dynamic Model

Many standard models for filtering and control assume the existence of a hidden state x_t that evolves following a Markov process through some state transition probability $p(x_{t+1}|x_t, u_t)$, where we made the dependency on the control action u_t explicit. The observations y_t are sampled from the hidden state x_t with some distribution $p(y_t|x_t)$, as described by the graphical model shown in **Figure 1**.

The fundamental assumption of this model is that there is a random variable of bounded complexity, the state x_t, that separates new observations y_{t+1} from all past ones y^t. The advantage of having such a variable is apparent in the classical filtering equations:

$$p(x_{t+1}|y^{t+1}, u^t) \propto p(y_{t+1}|x_{t+1}) \int p(x_{t+1}|x_t, u_t) p(x_t|y^t, u^{t-1}) \mathrm{d}x_t, \tag{5.}$$

$$p(y_{t+1}|y^t, u^t) = \int p(y_{t+1}|x_{t+1}) p(x_{t+1}|y^t, u^t) \mathrm{d}x_{t+1}. \tag{6.}$$

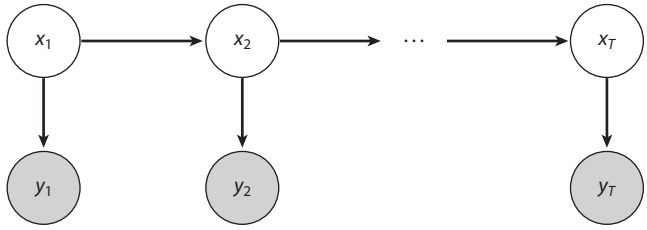

Figure 1

Graphical model of the evolution of the (hidden) state x_t and corresponding observations y_t. Notice that future observations are independent from past observations given the current state.

Here, all the information about the past data y^t is contained in the (relatively small) posterior $p(x_t|y^t)$. In this sense, the posterior is sufficient for the state update [i.e., for computing $p(x_{t+1}|y^{t+1}, u^t)$] and for the prediction of the data [i.e., for computing $p(y_{t+1}|y^t)$].

In a hidden Markov model or Kalman filter, the transitions are assumed to be linear, and the state and observations either Gaussian or discrete. In these cases, the posterior can be updated easily, and there are efficient algorithms to infer the model parameters of the system. However, for many real problems, the integrals in the filtering equation are not tractable since the transition operator is often nonlinear. In this case, the complexity of updating the posterior may grow exponentially (45). Furthermore, the data-generating distribution $p(y_{t+1}|x_{t+1})$ is difficult to compute or even to approximate. Finally, while we can always artificially ignore long-term dependencies and consider the system Markovian by augmenting the state $X_t = [x_{t-k}, \ldots, x_t]$, the resulting state may be too complex to handle.

While there is no obvious solution to these problems in general, it is often the case that we are interested not in predicting the data, but only in predicting the control action z, which can be quite different and far smaller than the data. In the next section, we see that this can guide the design of efficient filters.

6.2. Separating Representation

Rather than explicitly looking for a Markovian state that can generate the observed data y_t, i.e., inferring representations for prediction of the data, we focus on finding a representation (proxy state) x_t to predict a task variable z_t (for instance, a control input), which is generally far lower dimensional than the data and allows causal and recursive posterior updating using only the latest measurements. In this sense, this section is about inferring representations for control.

Motivated by Equation 5, we define the variable x_t through its posterior distribution $q(x_t|y^t, u^t)$,[5] and we require that it satisfies the following:

1. Prediction: The posterior of x_t is sufficient of y^t and u^t for z_{t+k}; that is, for each $0 \leq k < n$, we have

$$p(z_{t+k}|y^t, u^{t+k-1}) = \int q(z_{t+k}|x_t, u_t^{t+k-1}) q(x_t|y^t, u^{t-1}) \mathrm{d}x_t.$$

   This means that the distribution $q(x_t|y^t, u^{t-1})$ of the representation of past measurements and our model distribution $q(z_{t+k}|x_t, u_t^{t+k-1})$ that predicts the future task variable given the state are together sufficient to approximate the real posterior $p(z_{t+k}|y^t, u^{t+k-1})$ of the task given the past observations.

[5] We use q to distinguish the (unknown) data distribution p from our model distribution.

2. Update: the posterior of x_t is sufficient of y^t and u^t for x_{t+1}:

$$q(x_{t+1}|y^{t+1}, u^t) = \int q(x_{t+1}|x_t, y_{t+1}, u_t) q(x_t|y^t, u^{t-1}) \mathrm{d}x_t.$$

That is, the representation contains enough information about the dynamics to update itself given new measurements. For example, in a Kalman filter, the posterior of the position alone is not sufficient, since without knowing the speed, we cannot predict the future states.

Note that, like the classical filtering equations, this density propagation is exact. However, unlike the filtering equations, we can directly learn the transition probability $q(x_{t+1}|x_t, y_{t+1}, u_t)$ rather than use Bayes's rule, and therefore there is no need to compute the posterior $p(y_{t+1}|x_{t+1})$, which is generally intractable for high-dimensional and complex data such as natural images. The separator, in this case, is not the random variable x_t but the posterior density $q(x_t|y^t, u^t)$, which is in general infinite dimensional. As shown below, this model allows us to explicitly trade off the complexity of the representation with the fidelity of the separation.

Example 1 (Kalman filter). The method we propose reduces, in the linear Gaussian case, to the Kalman filter. Indeed, it does so in two different ways. First, let x_t be the state of a linear time-invariant Gaussian state-space model and let the task be one-step prediction, $z_t = y_{t+1}$. Now, let $\hat{x}_t$ be a random variable such that $q(\hat{x}_t|y^t) = p(x_t|y^t)$, where $p(\hat{x}_t|y^t)$ is the posterior computed by the Kalman filter, and let $q(z_t|\hat{x}_t) = p(y_{t+1}|x_t)$. Then, trivially, $p(z_t|y^t) = p(y_{t+1}|y^t) = \int p(y_{t+1}|x_t) p(x_t|y^t) \mathrm{d}x_t = \int q(y_{t+1}|\hat{x}_t) q(\hat{x}_t|y^t) \mathrm{d}x_t$, so the posterior computed by the Kalman filter is sufficient for predicting future measurements. Moreover, by letting $q(\hat{x}_{t+1}|\hat{x}_t, y_{t+1}) = p(x_{t+1}|x_t, y_{t+1}) = \frac{1}{Z} p(y_{t+1}|x_{t+1}) p(x_{t+1}|x_t)$, we see that $q(\hat{x}_t|y^t)$ is also sufficient for the update. Therefore, the posterior computed by the Kalman filter satisfies both the prediction and update models above. Notice, however, that this is not the only option. Instead, let $\hat{x}_t = (\hat{y}(y^t), P(y^t))$ be the mean and covariance of the innovation computed by the Kalman filter. Then x_t is a deterministic sufficient statistic (function of the past y^t). Notice that, in this case, the dimension of the representation $\hat{x}_t$ is larger, and the update equation is given by the more complex Riccati equation. Therefore, by adopting a deterministic representation, we have had to increase its computational complexity.

While in Equation 5 we need to use the prediction probability $p(y_t|x_t)$ to update the posterior, which is not tractable when the data y_t are high dimensional, using the prediction and update conditions given above, we have the simple iterative update rule

$$q(x_{t+1}|x_t, y^{t+1}, u^{t+1}) = \int q(x_{t+1}|x_t, y_t, u_t)\, q(x_t|y^t, u^t) \mathrm{d}x_t$$

and the task prediction rule

$$p(z_t|y^t, u^t) = q(z_t|y^t, u^t) = \int q(z_t|x_t) q(x_t|y^t, u^t) \mathrm{d}x_t,$$

where the first equality is due to the sufficiency hypothesis. Unlike Equation 5, these update equations involve only distributions over z_t and x_t, which are assumed to have lower effective dimension than the data y_t, or at least to have a simpler distribution (i.e., discrete or Gaussian).

Moreover, notice that if we restrict $q(x_t|y^t, u^t)$ to be degenerate (i.e., a Dirac delta), so that x_t is a deterministic function of the past history of the measurements, which this framework allows, then the integrals are trivial, and all updates can be computed exactly. On the other hand, allowing

a more complex form for $q(x_t|y^t, u^t)$ could drastically simplify the computation of both $q(z_t|x_t)$ and $q(x_{t+1}|y_{t+1}, u_t, x_t)$, so there is a trade-off between the cost of computing the integrals for $q(x_t|y^t, u^t)$ and the complexity of the prediction and update rules, as seen in the case of the Kalman filter. More specifically, when the model is linear and the driving input white, zero-mean Gaussian, and i.i.d., the posterior is Gaussian. Thus, one can consider either the posterior itself or the parameters that represent it (mean and covariance matrix) as the separator, with the latter being a deterministic representation.

While the complexity of a posterior $q(x_t|y^t, u^{t-1})$ sufficient for the task z_t is generally much smaller than what would be required to predict y_{t+1}, a representation x_t that satisfies all the required properties may still have a high dimension or high complexity. What we are after is an explicit way to trade off complexity with the quality of the representation, represented by its degree of sufficiency and Markovianity. As shown above in the static case, this trade-off can be expressed by the IBL, which is now

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}\sum_{k=0}^{n} H_{p,q}(z_{t+k}|y^t, u^{t+k-1}) + \beta I(x_t; y^t, u^t),$$

where $H_{p,q}$ is the cross-entropy between the real data distribution $p(z_t|y^t, u^t)$ and our model distribution $q(z_t|y^t, u^t) = \int q(z_t|x_t)q(x_t|y_t, u_t)\mathrm{d}x_t$ defined above.

Proposition 3 (*n*-step prediction loss). Given $q(z_t|x^t, u^t)$ and $q(x_t|y_t, u_t, x_{t-1})$, define $q(z_t|y^t) = \int q(z_t|x_t)q(x_t|y^t, u^t)\mathrm{d}x_t$ as above. Then the cross-entropy loss

$$\mathcal{L} = \frac{1}{T}\sum_{t=1}^{T}\sum_{k=0}^{n} H_{p,q}(z_{t+k}|y^t, u^{t+k-1})$$

is minimized if and only if the posterior $q(x_t|y^t)$ of x_t separates z_t from the past data y^t, meaning that $p(z_t|y^t, u^t) = \int q(z_t|x_t, u^t)q(x_t|y^t, u^t)\mathrm{d}x_t = F(q(x_t|y^t, u^t))$ for almost all y^t.

Proof. To simplify the notation, we consider only the case $n = 0$ (which corresponds to smoothing), the general case being identical. Recall that $H_{p,q}(z_t|y^t) = H_p(z_t|y^t) + \mathbb{E}_{y^t\sim p(y^t)}\mathrm{KL}(p(z_t|y^t) \,\|\, q(z_t|y^t))$, so

$$\begin{aligned}\mathcal{L} &= \frac{1}{T}\sum_{t=1}^{T} H_{p,q}(z_t|y^t)\\ &= \frac{1}{T}\sum_{t=1}^{T} H_p(z_t|y^t) + \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}_{y^t}\mathrm{KL}(p(z_t|y^t) \,\|\, q(z_t|y^t))\\ &\geq \frac{1}{T}\sum_{t=1}^{T} H_p(z_t|y^t).\end{aligned}$$

Since the degenerate representation $x_t = y^t$ trivially reaches the lower bound, for any representation minimizing the loss function, we must have $\mathbb{E}_{y^t}\mathrm{KL}(p(z_t|y^t) \,\|\, q(z_t|y^t)) = 0$ for all t and for y^t almost everywhere. In particular, for y^t almost everywhere, we have $p(z_t|y^t) = q(z_t|y^t) = \int q(z_t|x_t)q(x_t|y^t)\mathrm{d}x_t$. □

Remark 2. Notice that it is not the random variable x_t that separates z_t from y^t (i.e., $z_t \perp y^t|x_t$), as it was in the static case. Instead, it is its (posterior) distribution $q(x_t|y^t)$ that acts as the separator. However, if the latter is finitely parameterized [$q(x_t|y^t) = q_{\phi(y_t)}(x_t)$, where q_ϕ is a parameterized family of probability distributions and $\phi(y^t)$ is a function],

then $z_t \perp y^t | \phi(y^t)$; i.e., the parameters of the distributions can be interpreted as a finite-dimensional representation that separates the past data from the task.

Corollary 1. Suppose that there exists a separating variable x_t of finite complexity. Then, in the limit $\beta \to 0$, the IBL recovers a separating representation.

Proof. Since x_t has finite complexity, $\beta I(x_t; y^t) \to 0$ as $\beta \to 0$. Therefore, in the limit, the minimum of the Lagrangian is exactly

$$\mathcal{L}' = \frac{1}{T} \sum_{t=1}^{T} \sum_{k=0}^{n} H_{p,q}(z_{t+k} | y^t, u^{t+k-1}).$$

Therefore, any other minimizer x' of the IBL must also minimize $\mathcal{L}'$, and by the previous proposition, it must be a separating distribution. □

7. A SEPARATION PRINCIPLE FOR CONTROL

The previous section showed that, given a task z (a random variable to predict), it is possible to infer a representation x that trades off complexity with sufficiency and Markovianity. We now specialize this program for a control task, so that a controller operating on the representation behaves as if it had access to the entire past history of the data, analogously to the separation principle in LQG optimal control. Unlike LQG control, however, in general there is no finite-dimensional sufficient statistic, and therefore, following the program above, we seek a representation that trades off complexity with fidelity.

To this end, assume that our control task consists of minimizing a control loss R such that

$$R = \sum_{t=1}^{T} r_t(y^t, u^t),$$

where $r_t = r(y^t, u^t)$ is a possibly stochastic function of the true (global) state of the system x_t and the actions. Notice that even if the system is not Markovian, we can always assume that such a global state exists; in the worse case, $x_t = y^t$. Notice that LQG and other standard control losses can be written in this form. To simplify, we consider a finite horizon $T < \infty$.

We claim that if the posterior of x_t is a sufficient representation of the data y^t for the task $z_t = r_t$, then there exists an optimal control policy π' that is a function of the posterior $q(x_t | y^t, u^t)$ alone.

Proposition 4. Let x_t be such that the posterior $q_t = q(x_t | y^t, u^t)$ of x_t is sufficient of y^t, u^t for r_t, meaning that

$$p(r_{t+k} | y^t, u^{t+k}) = \int q(r_{t+k} | x_t, u_t^{t+k}) q(x_t | y^t, u^t) \mathrm{d}x_t.$$

Then there exists an optimal control policy $u_{t+1} = \pi^*(q_t)$ that minimizes the expected risk $\mathbb{E}[R|\pi^*]$ and depends on the past data y^t, u^t only through q_t.

Proof. Adopting standard reinforcement learning notation, let $Q^{\pi}_{>t}(y^t, u^t, u) = \mathbb{E}[R_{>t} | y^t, u_{t+1} = u, \pi]$ be the expected value of $R_{>t} = \sum_{t'=t}^{T} r_{t'}$ when following the policy π for the last $T - t$ steps given the observation history and action history y^t, u^t until now. Define the optimal Q-function $Q^*(y^t, u^t, u) = \max_{\pi} Q^{\pi}_{>t}(y^t, u^t, u)$.

Recall that, given $Q^*(y^t, u^t, u)$, the optimal policy is given by $\pi^*(y^t, u^t) = \operatorname{argmax}_u Q^*_{>t}(y^t, u^t, u)$. Therefore, to prove that the optimal policy depends only on q_t,

it suffices to prove that $Q^*_{>t}(y^t, u^t, u) = Q(q_t, u)$, i.e., that we can compute the optimal Q-function given q_t alone instead of the whole history y^t, u^t. This follows trivially from the fact that

$$Q^*_{>t}(y^t, u^t, u) = \min_{u_t^T : u_{t+1} = u} \sum_k^{T-t} \mathbb{E}[r_{t+k} | y^t, u^{t+k}]$$

$$= \min_{u_{t+1}^T : u_t = u} \int \left\{ \sum_{k=0}^{T-t} r_{t+k}\, q(r_{t+k} | x_t, u_t^{t+k}) \right\} \mathrm{d}q(x_t | y^t, u^t). \qquad \square$$

Notice that this proposition does not give an explicit way of learning a policy (since a naive application would require a brute force optimization over all possible actions). Rather, the usefulness of the theorem is that it proves that any representation sufficient to predict the rewards r_t, a fairly general condition, is also sufficient for control. In particular, if $r_t = r_t(m_t)$ is a function of some (low-dimensional) measurement m_t, as suggested by Dosovitskiy & Koltun (27), then a representation x_t trained to predict those measurements m_t will also be sufficient for r_t and hence for control. Dosovitskiy & Koltun (27) implicitly exploited this fact to learn a state-of-the-art control policy for a complex task and high-dimensional data. For example, suppose an agent whose task is to reach a location is provided with both a (high-dimensional) video input and a (low-dimensional) GPS input. Since the cost function (distance) can be expressed as a simple function of the low-dimensional measurements (i.e., the GPS position), a representation that is sufficient to predict future GPS measurements given the past input and the future control actions is also provably sufficient for optimal control. However, while training a representation directly for optimal control is difficult, training a representation in a supervised fashion to predict feature measurement is comparatively easy.

A more complex example is to learn to control an agent in a video game by predicting future numerical measurements provided by the game itself, such as score and remaining resources (lives or ammunition). For instance, Dosovitskiy & Koltun (27) assumed that the reward is a linear function of past measurements and therefore that future rewards can be computed from a prediction of the measurements, bypassing explicit computation of the Q-function of reinforcement learning and instead training to predict future measurements in a supervised fashion. Of course, in many cases it is not possible to write the reward as an explicit, let alone linear, function of the measurements. However, it may still be possible to express the reward as a function of a lower-dimensional random variable, as opposed to the hidden state of the environment. Any representation sufficient for this random variable is therefore a separating representation.

8. DISCUSSION

We have framed the problem of system identification for the purpose of control as that of inferring not deterministic statistics of sufficiently exciting time series, but rather an approximation of the posterior of the control loss given past measurements. While this approximation is in general infinite dimensional, universally approximating function classes, such as neural networks, can be employed in the inference. This approach yields some nice properties relative to classical Bayesian filtering. First, we do not need to apply Bayes's rule, and therefore there is no partition function to compute, to the benefit of computational complexity. Second, we do not need to make a strict assumption of Markovianity. Instead, we can explicitly trade off complexity with the fidelity of the approximation of the posterior. Such a posterior is the separator that plays the analogous role of the state of a Gaussian linear model in classical linear identification. The good news is that

the representations learned by generic SGD, while being agnostic of desirable properties of the resulting representation, end up enforcing them through implicit regularization, as we show for the static case.

Now for a few caveats. First, the representations we aim to infer are optimal when they are as good as the data for the chosen task. This does not mean they are good; if the data are uninformative (or insufficiently exciting), then there is no guarantee that can be made on the quality of the representation other than that it is sufficient, meaning that it is as good as the data (it can be no more, per the data-processing inequality). A completely independent problem is how to get as exciting data as possible, which is the problem of active learning or experiment design; this can be framed as an optimal control problem, which we do not address here.

Second, we are not suggesting that the model we propose is tractable in its most general form or that training a neural network to minimize the proposed IBL is easy. However, we show that minimizing a simple cross-entropy for a particular task (the control loss) leads to a representation that is sufficient for control. One should notice that this approach has strong links not only with the work of Dosovitskiy & Koltun (27) but also with reinforcement learning. Indeed, both can be seen as ways of making the algorithm tractable by directly approximating the expected loss for a given action.

More importantly, this class of tools opens several potentially exciting research avenues, both applied—making use of the power of these representations and implementing efficient algorithms to infer them—and theoretical, as little is known about the properties of these representations and their approximation bounds. This approach promises to reopen a field that has been shackled between the linear case (which is nice and elegant and for which a plethora of results are known, but which has very limited applicability) and the general case (where there is little to say and little that works in practice).

DISCLOSURE STATEMENT

The authors are not aware of any affiliations, memberships, funding, or financial holdings that might be perceived as affecting the objectivity of this review.

ACKNOWLEDGMENTS

Our research was supported by grants ONR N00014-17-1-2072, ARO W911NF-17-1-0304, and AFOSR FA9550-15-1-0229.

LITERATURE CITED

1. Kalman RE. 1960. A new approach to linear filtering and prediction problems. *ASME J. Basic Eng.* 82:35–45
2. Arun K, Kung S. 1990. Balanced approximation of stochastic systems. *SIAM J. Matrix Anal. Appl.* 11:42–68
3. Lindquist A, Picci G. 1979. On the stochastic realization problem. *SIAM J. Control Optim.* 17:365–89
4. Akaike H. 1974. A new look at the statistical model identification. *IEEE Trans. Autom. Control* 19:716–23
5. Lewis FL, Vrabie D, Syrmos VL. 2012. *Optimal Control*. New York: Wiley & Sons
6. Chiaromonte F, Cook RD, Li B. 2002. Sufficient dimension reduction in regressions with categorical predictors. *Ann. Stat.* 30:475–97
7. Shyr A, Urtasun R, Jordan MI. 2010. Sufficient dimension reduction for visual sequence classification. In *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3610–17. New York: IEEE
8. Sundaramoorthi G, Petersen P, Varadarajan VS, Soatto S. 2009. On the set of images modulo viewpoint and contrast changes. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 832–39. New York: IEEE

9. Krizhevsky A, Sutskever I, Hinton GE. 2012. ImageNet classification with deep convolutional neural networks. *Adv. Neural Inf. Process. Syst.* 25:1097–105
10. Cover TM, Thomas JA. 2012. *Elements of Information Theory*. New York: Wiley & Sons
11. Zhang C, Bengio S, Hardt M, Recht B, Vinyals O. 2016. Understanding deep learning requires rethinking generalization. arXiv:1611.03530
12. Achille A, Soatto S. 2018. On the emergence of invariance and disentangling in deep representations. *Int. J. Mach. Learn. Res.* In press
13. Jeffreys H. 1960. An extension of the Pitman–Koopman theorem. *Math. Proc. Camb. Philos. Soc.* 56:393–95
14. Bahadur RR. 1954. Sufficiency and statistical decision functions. *Ann. Math. Stat.* 25:423–62
15. Achille A, Soatto S. 2018. Information dropout: learning optimal representations through noisy computation. *IEEE Trans. Pattern Anal. Mach. Intell.* In press
16. Koren Y. 2010. Collaborative filtering with temporal dynamics. *Commun. ACM* 53:89–97
17. Wu CY, Ahmed A, Beutel A, Smola AJ, Jing H. 2017. Recurrent recommender networks. In *WSDM '17: Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 495–503. New York: ACM
18. Krishnan RG, Shalit U, Sontag D. 2015. Deep Kalman filters. arXiv:1511.05121
19. Raiko T, Tornio M. 2009. Variational Bayesian learning of nonlinear hidden state-space models for model predictive control. *Neurocomputing* 72:3704–12
20. Langford J, Salakhutdinov R, Zhang T. 2009. Learning nonlinear dynamic models. In *ICML '09: Proceedings of the 26th Annual International Conference on Machine Learning*, pp. 593–600. New York: ACM
21. Jazwinski AH. 2007. *Stochastic Processes and Filtering Theory*. North Chelmsford, MA: Courier
22. Wan EA, Van Der Merwe R. 2000. The unscented Kalman filter for nonlinear estimation. In *Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium*, pp. 153–58. New York: IEEE
23. Fox R, Tishby N. 2016. Minimum-information LQG control part II: retentive controllers. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 5603–9. New York: IEEE
24. Tiomkin S, Polani D, Tishby N. 2017. Control capacity of partially observable dynamic systems in continuous time. arXiv:1701.04984
25. Rubin J, Shamir O, Tishby N. 2012. Trading value and information in MDPS. In *Decision Making with Imperfect Decision Makers*, ed. TV Guy, M Kàrnỳ, DH Wolpert, pp. 57–74. Berlin: Springer
26. Fox R, Moshkovitz M, Tishby N. 2016. Principled option learning in Markov decision processes. arXiv:1609.05524
27. Dosovitskiy A, Koltun V. 2016. Learning to act by predicting the future. arXiv:1611.01779
28. Houthooft R, Chen X, Duan Y, Schulman J, De Turck F, Abbeel P. 2016. VIME: variational information maximizing exploration. *Adv. Neural Inf. Process. Syst.* 29:1109–117
29. Dong J, Soatto S. 2015. Domain-size pooling in local descriptors: DSP-SIFT. In *2015 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5097–106. New York: IEEE
30. Tishby N, Pereira FC, Bialek W. 1999. The information bottleneck method. In *The 37th Annual Allerton Conference on Communication, Control, and Computing*, ed. B Hajek, RS Sreenivas, pp. 368–77. Urbana: Univ. Ill.
31. Alemi AA, Fischer I, Dillon JV, Murphy K. 2016. Deep variational information bottleneck. arXiv:1612.00410
32. Soatto S. 2013. Actionable information in vision. In *Machine Learning for Computer Vision*, ed. R Cipolla, S Battiato, GM Farinella, pp. 17–48. Berlin: Springer
33. Strouse D, Schwab DJ. 2016. The deterministic information bottleneck. arXiv:1604.00268
34. LeCun Y, Boser B, Denker J, Henderson D, Howard R, et al. 1990. Handwritten digit recognition with a back-propagation network. *Adv. Neural Inf. Process. Syst.* 2:396–404
35. Soatto S, Chiuso A. 2016. Visual representations: defining properties and deep approximations. In *4th International Conference on Learning Representation (ICLR)*. **https://arxiv.org/abs/1411.7676**
36. Glorot X, Bengio Y. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 249–56. N.p.: PMLR

37. Nesterov Y. 2013. *Introductory Lectures on Convex Optimization: A Basic Course*. New York: Springer
38. Kingma D, Ba J. 2014. Adam: a method for stochastic optimization. arXiv:1412.6980
39. Mnih V, Kavukcuoglu K, Silver D, Rusu AA, Veness J, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518:529–33
40. Kingma DP, Salimans T, Welling M. 2015. Variational dropout and the local reparameterization trick. *Adv. Neural Inf. Process. Syst.* 28:2575–83
41. McAllester D. 2013. A PAC-Bayesian tutorial with a dropout bound. arXiv:1307.2118
42. Dziugaite GK, Roy DM. 2017. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Proceedings of the Thirty-Third Conference on Uncertainty in Artificial Intelligence*, chap. 173. N.p.: AUAI Press. **http://auai.org/uai2017/proceedings/papers/173.pdf**
43. Chaudhari P, Soatto S. 2017. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. arXiv:1710.11029
44. Hochreiter S, Schmidhuber J. 1997. Flat minima. *Neural Comput.* 9:1–42
45. Bar-Shalom Y, Fortmann TE. 1987. *Tracking and Data Association*. San Diego, CA: Academic