UNIVERSITY OF CALIFORNIA

Los Angeles

Emergent Properties of Deep Neural Networks

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Alessandro Achille

2019

ABSTRACT OF THE DISSERTATION

Emergent Properties
of Deep Neural Networks

by

Alessandro Achille
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2019
Professor Stefano Soatto, Chair

We show that information theoretic quantities can be used to control and describe the training process of Deep Neural Networks, and can explain how properties, such as invariance to nuisance variability and disentanglement of semantic factors, emerge naturally in the learned representation. Through its dynamics, stochastic gradient descent (SGD) implicitly regularizes the information in the weights, which can then be used to bound the generalization error through the PAC-Bayes bound. Moreover, the information in the weights can be used to defined both a topology and an asymmetric distance in the space of tasks, which can then be used to predict the training time and the performance on a new task given a solution to a pre-training task. While this information distance models difficulty of transfer in first approximation, we show the existence of non-trivial irreversible dynamics during the initial transient phase of convergence when the network is acquiring information, which makes the approximation fail. This is closely related to critical learning periods in biology, and suggests that studying the initial convergence transient can yield important insight beyond those that can be gleaned from the well-studied asymptotics.

The dissertation of Alessandro Achille is approved.

Alyson Fletcher

Stanley J Osher

Yingnian Wu

Stefano Soatto, Committee Chair

University of California, Los Angeles

2019

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, my thanks go to my advisor Stefano Soatto. This work would not exist without his unwavering support, patience and the many discussions. His own advisor, Pietro Perona, once remarked that the best way of teaching is leading through examples, and I am grateful to both of them for what they taught me, both as academics and as persons.

I was fortunate to collaborate with many gifted people: Irina Higgins, Tom Eccles, Loic Matthey, Chris Burgess, Nicholas Watters, Alexander Lerchner from DeepMind, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maki, Charless Fowlkes from AmazonAI. And special thanks go to some of my dearest friends, who also became collaborators: Glen Mbeng, Giovanni Paolini, Matteo Rovere.

UCLA and Caltech, which kindly hosted me for a term, have been welcoming and stimulating environments, and I am grateful to the people I met there. In particular, I would not have learned half as much without the neverending discussions with Pratik Chaudhari and Konstantin Tsotsos during (and after) my first years.

The material presented in this thesis has been disseminated through the following articles:

- Chapter 2: Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. doi: 10.1109/TPAMI.2017.2784440

- Chapter 3: Alessandro Achille, Tom Eccles, Loic Matthey, Chris Burgess, Nicholas Watters, Alexander Lerchner, and Irina Higgins. Life-long disentangled representation learning with cross-domain latent homologies. In *Advances in Neural Information Processing Systems 31*, pages 9895–9905. 2018

- Chapter 4: Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *Journal of Machine Learning Research*, 19(1): 1947–1980, 2018b; Alessandro Achille, Giovanni Paolini, Glen Mbeng, and Stefano Soatto. The Information Complexity of Learning Tasks, their Structure and their Distance. *arXiv:1904.03292*, 2019b

- Chapter 5: Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. Task2Vec: Task Embedding for Meta-Learning. *arXiv:1902.03545*, 2019a

- Chapter 6: Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep neural networks. *Proc. of the International Conference on Learning Representations (ICLR)*, 2019

I want to thank again my collaborators for making these papers possible: In their respective paper, Matteo Rovere has contributed equally in developing the ideas and designing the experiment, Irina Higgins proposed the topic and steered the paper with many insightful suggestions and analyses, and all coauthors contributed to various degrees with helpful suggestions, comments, discussions, and support in running the experiments.

This work as been supported with grants by ONR, AFOSR, ARO.

VITA

2010–2013 B.Sc. in Mathematics at Scuola Normale Superiore, Pisa.

2013–2015 M.Sc. in Pure Mathematics at Scuola Normale Superiore, Pisa.
Advisor: Alessandro Berarducci

2017 Research Scientist Intern at DeepMind. *Mentor:* Irina Higgins

2018 Applied Scientist Intern at AmazonAI. *Mentor:* Pietro Perona.

PUBLICATIONS

1. Alessandro Achille and Alessandro Berarducci. A vietoris-smale mapping theorem for the homotopy of hyperdefinable sets. *Selecta Mathematica*, pages 1–29, 4 2018. ISSN 1022-1824. doi: 10.1007/s00029-018-0413-3

2. Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. doi: 10.1109/TPAMI.2017.2784440

3. Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *Journal of Machine Learning Research*, 19(1):1947–1980, 2018b

4. Alessandro Achille and Stefano Soatto. A separation principle for control in the age of deep learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1), 2018a. doi: 10.1146/annurev-control-060117-105140

5. Alessandro Achille, Tom Eccles, Loic Matthey, Chris Burgess, Nicholas Watters, Alexander Lerchner, and Irina Higgins. Life-long disentangled representation learning with cross-domain latent homologies. In *Advances in Neural Information Processing Systems 31*, pages 9895–9905. 2018

6. Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. Task2Vec: Task Embedding for Meta-Learning. *arXiv:1902.03545*, 2019a

7. Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep neural networks. *Proc. of the International Conference on Learning Representations (ICLR)*, 2019

8. Alessandro Achille, Giovanni Paolini, Glen Mbeng, and Stefano Soatto. The Information Complexity of Learning Tasks, their Structure and their Distance. *arXiv:1904.03292*, 2019b

CHAPTER 1

Introduction

You may have seen tigers before, or at least pictures of them. You certainly have seen cats, likely in different sizes, poses, colors, from different vantage points, under different lighting conditions, etc. However, if you ever found yourself face-to-face with a feline trying to decide whether to run for your life, by the time you finish searching for similar images seen in the past, the decision may have been made for you. Despite having seen many images of cats and tigers, none may resemble the scene facing you. Hence, storing raw memories and searching through them to determine present danger does not seem like a viable survival strategy. What, then, to store in lieu of the data? How does the answer change if we need to solve more than one task? This work tries to establish a theory of representation learning and task learning to address these and other related questions.

When we say *task*, we mean generally a set of decisions we need to take given observed data. Discriminating cats from tigers is one. Survival is another. Retrieving the data from storage is yet another. Many tasks can be abstracted as determining the most likely value of a random variable (cat or tiger? alive or dead? same or not?) given some observed data.

A *representation for a task* is a function of the data that can be computed and stored in memory so that, when presented with new data sometime in the future, one can process it efficiently to accomplish a task. An optimal representation is one that enables accomplishing a task as well as possible given available data and resources.

The discussion so far entails, without mentioning them explicitly, two different but intertwined representations: One is a representation of past data about the task (*i.e.*, the training set). This is the outcome of learning, and, in a Deep Network Network, which will assume as

the reference model in this work,[1] can be identified to the information stored in the weights of the network. The other is a representation of a current inputs, which is instead encoded in the activations of the network.

The two attend different optimality criteria: Weights needs to store useful information that may be useful for taking future decision, without squandering resources in remembering spurious correlations or one-time events (Chapter 4). Furthermore, this information needs to be easily accessible at inference time. Activations, on the other end, needs to represent essential details of the scene at hand, without squandering resources on useless details, *e.g.*, representing tufts in the fur of the tiger (Chapter 2).

This dichotomy of representations creates a fundamental problem: Desirable properties of the activations concern the representation of future data. However, we can only optimize a representation of past data (weights). The relation between representations of past and future data goes to the heart of the issue of learning and generalization: How past experiences should shape the our perception of the world and create good representations that allow successfully solving the task (Chapter 4), and continue learning the solution to an ever-increasing set of tasks we may be interested in.

Information of what, for what, relative to what?

Data reduces uncertainty in the task. If we were to hear a roar, we would be more confident that the animal in front of us is a tiger, rather than a cat. The amount of uncertainty reduction is the *information* the data contains about that task. Reasoning about information requires specifying information *of what* (the data), *for what* (the task), *relative to what*ever prior uncertainty was present before the data was available. A representation should, ideally, be as informative for the task as *future* data will be, while being invariant to nuisance factors affecting it. Of course, we do not have access to future data. We do, however, have access to *past* data. How informative is the memory of past data for the purpose of making future

[1]Most of theory is general and does not depend on the specific model, expect for the results in Chapter 4 connecting invariance of the learned representation to the complexity of the weights.

decisions? How much information is stored in memory? That depends on how we measure uncertainty and there are many alternate ways of doing so. We contend in this paper that what matters is the uncertainty *of what* we are computing, rather than how we compute it. The central thesis is that the quantity that matters for deep learning is the information in the weights.

CHAPTER 2

Information in the Activations

By the Data Processing Inequality, no function of the data (representation) can be better than the data themself for decision and control (task). However, most organisms and algorithms use complex representations that deeply alter the input in order to solve the task. Even in contemporary deep networks, of the three main three type of operation used – convolutions, ReLU non-linearities and max-pooling – the only effect of ReLUs and max-pooling is to throw away information.

In fact, as we will see in this and the following chapters, keeping all information is far from optimal in a learning context. Assume, for example, that our task is learning to predict a label y given an input image x. By the curse of dimensionality, the number of samples would grow exponentially with the number of dimensions, so that for a relatively low-resolution 256×256 input, we would need $\sim 10^{28462}$ samples. Then, how can we learn at all on images?

Fortunately, both the distribution $p(x)$ of natural images and and the usual task distributions $p(y|x)$ are far from being generic distributions, for which the worst case bound would apply. In particular, three properties can be exploited:

1. **Nuisance invariance**, *i.e.* reduced dimension of the input space;
2. **Compositionally**, *i.e.* reduced dimension of the representation space (Chapter 3);
3. **Complexity prior** on the solution, *i.e.* reduced dimension of hypothesis space (Chapter 4).

This chapter concern the former approach. Nuisances are factors of variation that affect the data, but are otherwise irrelevant for the task. For example, change of illuminations,

point of view, shadows, occlusions, can change the input image, but often do not change the inference. As it turns out, most of the information in an image is due to nuisance variability (Sundaramoorthi et al., 2009), hence a good representation should collapse images differing only for nuisance variability, so to decrease the effective dimensionality of input space and reduce sample complexity.

Historically, this problem has been tackled by modeling nuisances as group actions, and finding the respective quotient space in order to obtain a group invariant representation. This approach however is difficult to carry, as most nuisances correspond to either complex groups (*e.g.*, the effect of a view-point change is a general diffeomorphism), or cannot be modeled as groups (occlusions). In this chapter, we introduce a generalization of invariance in information theoretic terms, and derive a variational loss function that can be used to test and enforce invariance of the representation. Interestingly, this loss coincides with the Information Bottleneck loss that Tishby et al. (1999) introduced as a generalization of minimal sufficient statistcs. Moreover, it can be interpreted as a regularized version of the standard cross-entropy loss used in Deep Learning. Using these results, we design an algorithm to explicitly minimize the loss, which we call Information Dropout, and has a generalized version of Variational Auto-Encoders as a particular case. Finally, we show that, as a side effect of introducing noise in the network, we can bias the network toward learning invariant and disentangled representations. However, the framework has an important drawback in that it gives a loss function that can be used to test optimality of the representation on test data, but does not speak of what is the optimal use of the training data, which will be instead the focus of next chapter.

2.1 Preliminaries

In the general supervised setting, we want to learn the conditional distribution $p(y|x)$ of some random variable y, which we refer to as the *task*, given some input data x. In typical applications, x is often high dimensional (for example an image or a video), while y is low dimensional, such as a label or a coarsely-quantized location.

We say that z is a **representation** of x if z is a stochastic function of x, or equivalently if the distribution of z is fully described by the conditional $p(z|x)$. In particular we have the Markov chain $y \to x \to z$. We say that a representation z of x is **sufficient** for y if $y \perp\!\!\!\perp x \mid z$, or equivalently if $I(z;y) = I(x;y)$; it is **minimal** when $I(x;z)$ is smallest among sufficient representations.

A **nuisance** is any random variable that affects the observed data x, but is not informative to the task we are trying to solve. More formally, a random variable n is a nuisance for the task y if $y \perp\!\!\!\perp n$, or equivalently $I(y;n) = 0$. Similarly, we say that the representation z is **invariant** to the nuisance n if $z \perp\!\!\!\perp n$, or $I(z;n) = 0$. When z is not strictly invariant but it minimizes $I(z;n)$ among all sufficient representations, we say that the representation z is **maximally insensitive** to n.

One typical example of nuisance is a group G, such as translation or rotation, acting on the data. In this case, a deterministic representation f is invariant to the nuisances if and only if for all $g \in G$ we have $f(g \cdot x) = f(x)$. Our definition however is more general in that it is not restricted to deterministic functions, nor to group nuisances.

An important consequence of the more general definition of nuisances is the following result, which shows that the observed data x can always be written as a deterministic function of the task y and of all nuisances n affecting the data.

Proposition 2.1.1 (Task-nuisance decomposition)**.** *Given a joint distribution $p(x, y)$, where y is a discrete random variable, we can always find a random variable n independent of y such that $x = f(y, n)$, for some deterministic function f.*

Proof. Let $n \sim \text{Unif}(0, 1)$ be sampled from the uniform distribution on $[0, 1]$. We claim that, for a fixed value of y, there is a function $\Phi_y(n)$ such that $x|y = \Phi_{y*}(n)$, where $(\cdot)_*$ denotes the push-forward map of measures. Given the claim, let $\Phi(y, n) = (y, \Phi_y(n))$. Since y is a discrete random variable, $\Phi(y, n)$ is easily seen to be a measurable function and by construction $(x, y) \sim \Phi_*(y, n)$. To see the claim, notice that, since there exists a measurable isomorphism between $\mathbb{R}^n$ and $\mathbb{R}$ (Theorem 3.1.1 of Berberian (1988)), we can assume without loss of generality that $x \in \mathbb{R}$. In this case, by definition, we can take $\Phi_y(n) = F_y^{-1}(n)$ where

$F_y(t) = \mathbb{P}[x < t \,|\, y]$ is the cumulative distribution function of $p(x|y)$. □

2.2 What is an optimal representation?

To simplify the inference process, instead of working directly with the observed high dimensional data x, we want to use a representation z that captures and exposes only the information relevant for the task y. Ideally, such a representation should be:

(i) **sufficient** for the task y, that is, we want $I(y; z) = I(y; x)$, so that information about y is not lost. Equivalently, we ask that the Markov chain $y \to z \to x$ holds;

(ii) **minimal** among all sufficient representation, that is, we want $I(z; x)$ to be minimal, so that it retains as little about x as possible, simplifying the role of the classifier;

(iii) **invariant** to the effect of nuisances $I(z; n) = 0$, so that the final classifier will not overfit to spurious correlations present in the training dataset between nuisances n and labels y;

(iv) maximally **disentangled**: such a representation, if it exists, would not be unique, since any bijective mapping preserves all these properties. We can use this to further aim to make the representation maximally disentangled, *i.e.,* choose the one(s) for which $\mathrm{TC}(z)$ is minimal. This simplifies the classifier rule, since no information will be present in the higher-order correlations between the components of z.

Finding a representation satisfying all these properties at the same time may seem a daunting tasks, so we will first focus on two of these properties, minimality and sufficiency, which can easily be written as a constrained optimization problem. We will then show that invariance and disentanglement are implied, or can easily be obtained by simply imposing minimality of the representation.

2.3 A Variational Principle for Learning Invariant Representations

Let's consider first the problem of finding minimal sufficient representations. The two conditions (i) and (ii) together are equivalent to finding a distribution $p(\mathbf{z}|\mathbf{x})$ which solves the constrained optimization problem

$$\begin{aligned} \text{minimize} \quad & I(\mathbf{x};\mathbf{z}) \\ \text{s.t.} \quad & I(\mathbf{x};\mathbf{y}) = I(\mathbf{z};\mathbf{y}). \end{aligned}$$

However, this minimization remains difficult in general due to the complexity of the nonlinear constrain. To address this problem, Tishby et al. (1999) considered the relaxed minimization problem given by the corresponding Lagrangian:

$$\mathcal{L} = I(x;y) - I(y;z) + \beta I(x;z),$$

which, using the identity $I(y;z) = H(y) - H(y|z)$, and using the fact that $I(x;y)$ and $H(y)$ do not depend on $p(z|x)$, we can further simplify as

$$\mathcal{L} = H(\mathbf{y}|\mathbf{z}) + \beta I(\mathbf{x};\mathbf{z}). \tag{2.1}$$

Here β is a positive constant that manages the trade-off between sufficiency (the performance on the task, as measured by the first term) and minimality (the complexity of the representation, measured by the second term). It is easy to see that, in the limit $\beta \to 0^+$, this is equivalent to the original problem, where $\mathbf{z}$ is a minimal sufficient statistic.

When all random variables are discrete and $\mathbf{z} = T(\mathbf{x})$ is a deterministic function of $\mathbf{x}$, the algorithm proposed by Tishby et al. (1999) can be used to minimize the IB Lagrangian efficiently. However, no algorithm is known to minimize the IB Lagrangian for non-Gaussian, high-dimensional continuous random variables. One of our key results is that, when we restrict to the family of distributions obtained by injecting noise to one layer of a neural network, we can efficiently approximate and minimize the IB Lagrangian.[1] As we will show,

[1]Since we restrict the family of distributions, there is no guarantee that the resulting representation will be optimal. We can, however, iterate the process to obtain incrementally improved approximations.

this process can be effectively implemented through a generalization of the dropout layer that we call *Information Dropout*.

To set the stage, we rewrite the IB Lagrangian as a per-sample loss function. Let $p(\mathbf{x}, \mathbf{y})$ denote the true distribution of the data, from which the training set $\{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1,\dots,N}$ is sampled, and let $p_\theta(\mathbf{z}|\mathbf{x})$ and $p_\theta(\mathbf{y}|\mathbf{z})$ denote the unknown distributions that we wish to estimate, parametrized by θ. Then, we can write the two terms in the IB Lagrangian as

$$H(\mathbf{y}|\mathbf{z}) \simeq \mathbb{E}_{\mathbf{x},\mathbf{y}\sim p(\mathbf{x},\mathbf{y})}\left[\mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x})}[-\log p_\theta(\mathbf{y}|\mathbf{z})]\right]$$
$$I(\mathbf{x};\mathbf{z}) = \mathbb{E}_{\mathbf{x}\sim p(\mathbf{x})}[\mathrm{KL}(p_\theta(\mathbf{z}|\mathbf{x}) \parallel p_\theta(\mathbf{z}))],$$

where KL denotes the Kullback-Leibler divergence. We can therefore approximate the IB Lagrangian empirically as

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{z\sim p(\mathbf{z}|\mathbf{x}_i)}[-\log p(\mathbf{y}_i|\mathbf{z})] + \beta\,\mathrm{KL}(p_\theta(\mathbf{z}|\mathbf{x}_i) \parallel p_\theta(\mathbf{z})). \tag{2.2}$$

Notice that the first term simply is the average cross-entropy, which is the most commonly used loss function in deep learning. The second term can then be seen as a regularization term. In fact, many classical regularizers, like the L_2 penalty, can be expressed in the form of eq. (2.2) (see also Gal and Ghahramani (2015)). In this work, we interpret the KL term as a regularizer that penalizes the transfer of information from $\mathbf{x}$ to $\mathbf{z}$. In the next section, we discuss ways to control such information transfer through the injection of noise.

Remark (Deterministic vs. stochastic representations)**.** Aside from being easier to work with, stochastic representations can attain a lower value of the IB Lagrangian than any deterministic representation. For example, consider the task of reconstructing single random bit y given a noisy observation x. The only deterministic representations are equivalent to the either the noisy observation itself or to the trivial constant map. It is not difficult to check that for opportune values of β and of the noise, neither realize the optimal tradeoff reached by a suitable stochastic representation.

Remark (Approximate sufficiency)**.** The quantity $I(x;y|z) = H(y|z) - H(y|x) \geq 0$ can be seen as a measure of the distance between $p(x,y,z)$ and the closest distribution $q(x,y,z)$

such that $\mathbf{x} \to \mathbf{z} \to \mathbf{y}$ is a Markov chain. Therefore, by minimizing eq. (2.1) we find representations that are increasingly "more sufficient", meaning that they are closer to an actual Markov chain.

2.4 Minimal if and only if invariant

The IB Lagrangian in eq. (2.1) only enforces (a trade-off between) sufficiency and minimality of the representation. What about the other properties? In this section we show that we only need to enforce (a) sufficiency and (b) minimality, from which invariance and disentanglement follow naturally thanks to the stacking of noisy layers of computation in deep networks. We will then show that sufficiency and minimality of the learned representation can also be promoted easily through implicit or explicit regularization during the training process.

Proposition 2.4.1 (Invariance and minimality)**.** *Let n be a nuisance for the task y and let z be a sufficient representation of the input x. Suppose that z depends on n only through x (*i.e., *$n \to x \to z$). Then,*

$$I(z;n) \leq I(z;x) - I(x;y).$$

Moreover, there is a nuisance n such that equality holds up to a (generally small) residual ϵ

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

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

Remark 2.4.2. Since $\epsilon \leq H(y|x)$, and usually $H(y|x) = 0$ or at least $H(y|x) \ll I(x;z)$, we can generally ignore the extra term.

Proof. By hypothesis, we have the Markov chain $(y, n) \to x \to z$; therefore, by the DPI, we have $I(z;y,n) \leq I(z;x)$. The first term can be rewritten using the chain rule as $I(z;y,n) = I(z;n) + I(z;y|n)$, giving us

$$I(z;n) \leq I(z;x) - I(z;y|n).$$

Now, since y and n are independent, $I(z;y|n) \geq I(z;y)$. In fact,

$$\begin{aligned} I(z;y|n) &= H(y|n) - H(y|z,n) \\ &= H(y) - H(y|z,n) \\ &\geq H(y) - H(y|z) = I(y;z). \end{aligned}$$

Substituting in the inequality above, and using the fact that z is sufficient, we finally obtain

$$I(z;n) \leq I(z;x) - I(z;y) = I(z;x) - I(x;y).$$

Moreover, let n be as in Lemma 2.1.1. Then, since x is a deterministic function of y and n, we have

$$I(z;x) = I(z;n,y) = I(z;n) + I(z;y|n),$$

and therefore

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

with ϵ defined as above. Using the sufficiency of z, the previous inequality for $I(z;y|n)$, the DPI, we get the chain of inequalities

$$\begin{aligned} \epsilon &= I(z;y|n) - I(x;z) \leq I(x;y|n) - I(x;y) \\ &\leq H(y|n) - H(y|n,z) - H(y) + H(y|x) \\ &\leq H(y) - H(y|n,z) - H(y) + H(y|x) \\ &= H(y|x) - H(y|n,z) \\ &\leq H(y|x) \end{aligned}$$

from which we obtain the desired bounds for ϵ. □

An important consequence of this proposition is that we can construct invariants by simply reducing the amount of information z contains about x, while retaining the minimum amount $I(z;x)$ that we need for the task y. This provides the network a way to automatically learn invariance to complex nuisances, which is complementary to the invariance imposed by the architecture. Specifically, one way of enforcing minimality explicitly, and hence invariance, is through the IB Lagrangian.

Corollary 2.4.3 (Invariants from the Information Bottleneck)**.** *Minimizing the IB Lagrangian*

$$\mathcal{L}(p(z|x)) = H(y|z) + \beta\, I(z;x),$$

in the limit $\beta \to 0$, yields a sufficient invariant representation z of the test datum x for the task y.

Hence, remarkably, the IB Lagrangian can be seen as the standard cross-entropy loss, plus a regularizer $I(z;x)$ that promotes invariance, and we indeed we will see how this can be done explicitly in a DNN. However, in addition to modifying the cost function, this shows that invariance can also be fostered implicitly by choice of architecture:

Corollary 2.4.4 (Bottlenecks promote invariance)**.** *Suppose we have the Markov chain of layers*

$$x \to z_1 \to z_2,$$

and suppose that there is a communication or computation bottleneck between z_1 and z_2 such that $I(z_1;z_2) < I(z_1;x)$. Then, if z_2 is still sufficient, it is more invariant to nuisances than z_1. More precisely, for all nuisances n we have $I(z_2;n) \leq I(z_1;z_2) - I(x;y)$.

Such a bottleneck can happen for example because $\dim(z_2) < \dim(z_1)$, *e.g.*, after a pooling layer, or because the channel between z_1 and z_2 is noisy, *e.g.*, because of dropout.

Proposition 2.4.5 (Stacking increases invariance)**.** *Assume that we have the Markov chain of layers*

$$x \to z_1 \to z_2 \to \ldots \to z_L,$$

and that the last layer z_L is sufficient of x for y. Then z_L is more insensitive to nuisances than all the preceding layers.

Notice, however, that the above corollary does not simply imply that the more layers the merrier, as it assumes that one has successfully trained the network (z_L is sufficient), which becomes increasingly difficult as the size grows. Also note that in some architectures, such as ResNets (He et al., 2016), the layers do not necessarily form a Markov chain because of skip connections; however, their "blocks" still do.

2.5 Learning disentangled representations

In addition to sufficiency and minimality, "disentanglement of hidden factors" is often cited as a desirable property of a representation (Bengio et al., 2013), but seldom formalized. We may think that the observed data is generated by a complex interplay of independent causes, or factors. Ideally, the components of the learned representation should capture these independent factors by disentangling the correlations in the observed data. We can then quantify disentanglement by measuring the *total correlation* (Watanabe, 1960), also known as *multiinformation* (Studenỳ and Vejnarová, 1998),[2] defined as

$$\mathrm{TC}(\mathbf{z}) := \mathrm{KL}(q(\mathbf{z}) \parallel \textstyle\prod_j q_j(z_j)).$$

Notice that the components of $\mathbf{z}$ are mutually independent if and only if $\mathrm{TC}(z)$ is zero. Adding this as a penalty in the IB Lagrangian, with a factor γ yields

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{z\sim p(\mathbf{z}|\mathbf{x}_i)}[-\log p(\mathbf{y}_i|\mathbf{z})] + \beta\,\mathrm{KL}(p_\theta(\mathbf{z}|\mathbf{x}_i) \parallel p_\theta(\mathbf{z})) + \gamma\,\mathrm{TC}(\mathbf{z}). \tag{2.3}$$

In general, minimizing this augmented loss is intractable, since to compute both the KL term and the total correlation, we need to know the marginal distribution $p_\theta(z)$, which is not easily computable. However, the following proposition, that we prove in **??**, shows that if we choose $\gamma = \beta$, then the problem simplifies, and can be easily solved by adding an auxiliary variable.

Proposition. *Let $\mathbf{z} = (z_1, \ldots, z_n)$ be a discrete random variable (the continuous case is analogous), let $p(\mathbf{z}|\mathbf{x})$ be a generic probability distribution, and let $q(\mathbf{z}) = \prod_{i=1}^{n} q_i(z_i)$ be a factorized prior distribution. Then, for any function $F(p)$, a minimization problem in the form*

$$\text{minimize}_{p,q} \quad F(p) + \beta\mathbb{E}_x[\mathrm{KL}(p(\mathbf{z}|\mathbf{x}) \parallel q(\mathbf{z}))],$$

[2] Multi-information would be a more common name for this quantity: We chose to use Total Correlation both for historical reasons, after its introduction by Watanabe (1960), and to emphasize the relation with disentanglement also in recent work on unsupervised learning (Steeg, 2017). Other measures of independence are of course possible. Total Correlation has the advantage of being enforced naturally when optimizing other information-theoretic quantities.

is equivalent to

$$\text{minimize}_p \quad F(p) + \beta \left\{ I_p(\mathbf{z}; \mathbf{x}) + \text{TC}_p(\mathbf{z}) \right\},$$

where $I_p(\mathbf{z}; \mathbf{x})$ *is the mutual information and* $\text{TC}_p(\mathbf{z})$ *is the total correlation of* $\mathbf{z}$*, assuming* $\mathbf{z} \sim p(\mathbf{z})$*.*

Proof. First, notice that for any $p(\mathbf{z}|\mathbf{x})$ we have

$$\text{KL}(p(\mathbf{z}) \parallel \textstyle\prod_i p(z_i)) = \min_{q(z_1),\dots,q(z_n)} \mathbb{E}\Big[\log \frac{p(\mathbf{z}|\mathbf{x})}{\prod_i q(z_i)} \Big],$$

since for any $p(\mathbf{z})$ and $q(\mathbf{z}) = \prod_i q(z_i)$ we have

$$\begin{aligned}
\text{KL}(p(\mathbf{z}) \parallel \textstyle\prod_i p(z_i)) &= \mathbb{E}\left[\log \frac{p(\mathbf{z})}{\prod_i p(z_i)} \right] = \mathbb{E}\left[\log \frac{p(\mathbf{z})}{\prod_i q(z_i)} \cdot \frac{\prod_i q(z_i)}{\prod_i p(z_i)} \right] \\
&= \mathbb{E}\left[\log \frac{p(\mathbf{z})}{\prod_i q(z_i)} \right] - \sum_{i=1}^{n} \text{KL}(p(z_i) \parallel q(z_i)) \\
&\leq \mathbb{E}\left[\log \frac{p(\mathbf{z})}{\prod_i q(z_i)} \right],
\end{aligned}$$

and equality trivially holds when $q(z_i) = p(z_i)$. Using the above identity, we now have

$$\begin{aligned}
I(\mathbf{x}; \mathbf{z}) + \text{KL}(p(\mathbf{z}|\mathbf{x}) \parallel \textstyle\prod_i p(\mathbf{z}_i|x)) &= \min_{q(z_1),\dots,q(z_n)} \mathbb{E}\Big[\log \frac{p(\mathbf{z}|\mathbf{x})}{p(\mathbf{z})} \cdot \frac{p(\mathbf{z})}{\prod_i q(z_i)} \Big] \\
&= \min_{q(z_1),\dots,q(z_n)} \mathbb{E}\left[\log \frac{p(\mathbf{z}|\mathbf{x})}{\prod_i q(z_i)} \right]. \qquad \square
\end{aligned}$$

In other words, minimizing the standard IB Lagrangian assuming that the activations are independent, *i.e.* having $q(\mathbf{z}) = \prod_i q_i(\mathbf{z}_i)$, is equivalent to enforcing disentanglement of the hidden factors. It is interesting to note that this independence assumption is already adopted often by practitioners on grounds of simplicity, since the actual marginal $p(\mathbf{z}) = \int_{\mathbf{x}} p(\mathbf{x}, \mathbf{z}) d\mathbf{x}$ is often incomputable. That using a factorized model results in "disentanglement" was also observed empirically by Higgins et al. (2017b), and we use it extensively in Chapter 3.

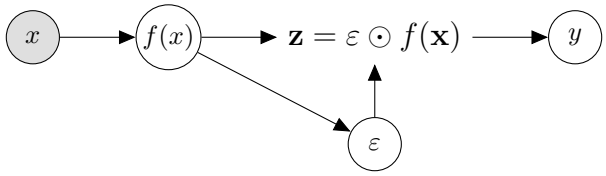
In view of the previous proposition, from now on we will assume that the activations are independent and ignore the total correlation term.

2.6 Learning optimal representations through noisy computations

Guided by the analysis in the previous sections, and to emphasize the role of stochasticity, we consider representations $\mathbf{z}$ obtained by computing a deterministic map $f(\mathbf{x})$ of the data (for instance a sequence of convolutional and/or fully-connected layers of a neural network), and then multiplying the result component-wise by a random sample ϵ drawn from a parametric noise distribution p_α with unit mean and variance that depends on the input $\mathbf{x}$:

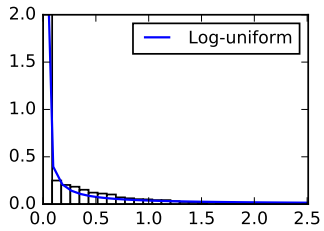
$$\varepsilon \sim p_{\alpha(\mathbf{x})}(\varepsilon),$$
$$\mathbf{z} = \varepsilon \odot f(\mathbf{x}),$$

where "$\odot$" denotes the element-wise product. Notice that, if $p_{\alpha(\mathbf{x})}(\varepsilon)$ is a Bernoulli distribution rescaled to have mean 1, this reduces exactly to the classic binary dropout layer. As we discussed in Section 2.10, there are also variants of dropout that use different distributions.

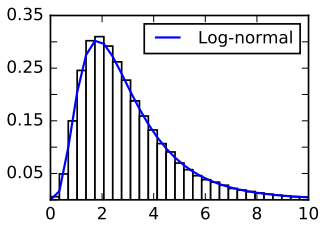


A natural choice for the distribution $p_{\alpha(\mathbf{x})}(\varepsilon)$, which also simplifies the theoretical analysis, is the log-normal distribution $p_{\alpha(\mathbf{x})}(\varepsilon) = \log \mathcal{N}(0, \alpha_\theta^2(\mathbf{x}))$. Once we fix this noise distribution, given the above expression for $\mathbf{z}$, we can easily compute the distribution $p_\theta(\mathbf{z}|\mathbf{x})$ that appears in eq. (2.2). However, to be able to compute the KL-divergence term, we still need to fix a prior distribution $q_\theta(\mathbf{z})$. The choice of this prior largely depends on the expected distribution of the activations $f(\mathbf{x})$. Recall that, by Section 2.5, we can assume that all activations are independent, thus simplifying the computation. Now, we concentrate on two of the most common activation functions, the *rectified linear unit* (ReLU), which is easy to compute and works well in practice, and the *Softplus* function, which can be seen as a strictly positive and differentiable approximation of ReLU.

A network implemented using ReLUs and batch normalization has the remarkable property of being scale-invariant, meaning that multiplying all weights, biases, and activations



(a) Histogram of ReLU activations



(b) Histogram of Softplus activations

Figure 2.1: Comparison of the empirical distribution $p(z)$ of the post-noise activations with our proposed prior when using: (a) ReLU activations, for which we propose a log-uniform prior, and (b) Softplus activations, for which we propose a log-normal prior. In both cases, the empirical distribution approximately follows the proposed prior. Both histograms where obtained from the last dropout layer of the All-CNN-32 network described in **??**, trained on CIFAR-10.

by a constant does not change the final result. Therefore, from a theoretical point of view, it would be desirable to use a scale-invariant prior. The only such prior is the improper log-uniform, $q(\log(z)) = c$, or equivalently $q(z) = c/z$, which was also suggested by Kingma et al. (2015), but as a prior for the weights of the network, rather than the activations. Since the ReLU activations are frequently zero, we also assume $q(z = 0) = q_0$ for some constant $0 \leq q_0 \leq 1$. Therefore, the final prior has the form $q(z) = q_0\delta_0(z) + c/z$, where δ_0 is the Dirac delta in zero. In Figure 2.1a, we compare this prior distribution with the actual empirical distribution $p(z)$ of a network with ReLU activations.

In a network implemented using Softplus activations, a log-normal is a good fit of the distribution of the activations. This is to be expected, especially when using batch-normalization, since the pre-activations will approximately follow a normal distribution with zero mean, and the Softplus approximately resembles a scaled exponential near zero. Therefore, in this case we suggest using a log-normal distribution as our prior $q(z)$. In Figure 2.1b, we compare this prior with the empirical distribution $p(z)$ of a network with Softplus activations.

Using these priors, we can finally compute the KL divergence term in eq. (2.2) for both

ReLU activations and Softplus activations.

Proposition 2.6.1 (Information dropout cost for ReLU)**.** *Let $z = \varepsilon \cdot f(x)$, where $\varepsilon \sim p_\alpha(\varepsilon)$, and assume $p(z) = q\delta_0(z) + c/z$. Then, assuming $f(x) \neq 0$, we have*

$$\mathrm{KL}(p_\theta(z|x) \parallel p(z)) = -H(p_{\alpha(x)}(\log \varepsilon)) + \log c$$

In particular, if $p_\alpha(\varepsilon)$ is chosen to be the log-normal distribution $p_\alpha(\varepsilon) = \log \mathcal{N}(0, \alpha_\theta^2(x))$, we have

$$\mathrm{KL}(p_\theta(z|x) \parallel p(z)) = -\log \alpha_\theta(x) + const. \tag{2.4}$$

If instead $f(x) = 0$, we have

$$\mathrm{KL}(p_\theta(z|x) \parallel p(z)) = -\log q.$$

Proof. Since the measures $P_\theta(z|x)$ and $P_\theta(z)$ are not absolutely continuous with respect to the Lebesgue measure, it will be convenient to use the more general definition of KL divergence $\mathrm{KL}(P(z) \parallel Q(z)) := \int \log \frac{dP}{dQ} dP$, where we assume that $P \ll Q$, so that the density dP/dQ exists. Recall that the KL-divergence is invariant under invertible parameter transformations, that is, $\mathrm{KL}(P(z) \parallel Q(z)) = \mathrm{KL}(P(\psi(z)) \parallel Q(\psi(z))$ for any invertible transformation $\psi(z)$. Let's first consider the case $f(x) \neq 0$, in which case we also have $z \neq 0$ almost surely. Since $p(\log z) = c$ for $z > 0$, we can write

$$\frac{dP(\log z|x)}{dP(\log z)} = p_{\alpha(x)}(\log \epsilon)/c.$$

Therefore, we have

$$\begin{aligned}
\mathrm{KL}(P_\theta(z|x) \parallel P_\theta(z)) &= \mathrm{KL}(P_\theta(\log z|x) \parallel P_\theta(\log z)) \\
&= \int \log \left(\frac{dP_\theta(\log z|x)}{dP_\theta(\log z)} \right) p_\theta(\log z|x) dz \\
&= \int \log \left(p_{\alpha(x)}(\log \varepsilon) \right) p_{\alpha(x)}(\log \varepsilon) d\varepsilon - \log c \\
&= -H(p_{\alpha(x)}(\log \varepsilon)) - \log c,
\end{aligned}$$

which gives us the first statement. Now, notice that if $p_\alpha(z)(\epsilon) = \log \mathcal{N}(0, \alpha_\theta^2(x))$, then by definition we have $p_{\alpha(x)}(\log \epsilon) = \mathcal{N}(0, \alpha_\theta^2(x))$. We can then use the known formula for the

entropy of a Gaussian distribution to obtain

$$H(\mathcal{N}(0,\alpha)) = \log \alpha_\theta(x) + \frac{1}{2}\log(2\pi e),$$

which gives us the second statement.

Finally, consider the case $f(x) = 0$, in which case $z = 0$, and therefore $p(z|x)$ reduces to the singular distribution $p(z|x) = \delta_0(z)$. Again, the measure $P(z|x)$ is absolutely continuous with respect to $P(z)$, and we have $dP(z|x)/dP(z) = 1/q$ if $z = 0$ and $dP(z|x)/dP(z) = 0$ if $z \neq 0$. We then readily have $\mathrm{KL}(p_\theta(z|x) \parallel p(z)) = \int \log \frac{dP(z|x)}{dP(z)} dP = -\log q$. □

Proposition 2.6.2 (Information dropout cost for Softplus)**.** *Let $z = \varepsilon \cdot f(x)$, where $\varepsilon \sim p_\alpha(\varepsilon) = \log \mathcal{N}(0, \alpha_\theta^2(x))$, and assume $p_\theta(z) = \log \mathcal{N}(\mu, \sigma^2)$. Then, we have*

$$\mathrm{KL}(p_\theta(z|x) \parallel p(z)) = \frac{1}{2\sigma^2}\left(\alpha^2(x) + \mu^2\right) - \log\frac{\alpha(x)}{\sigma} - \frac{1}{2}. \tag{2.5}$$

Proof. Since the KL-divergence is invariant for invertible reparametrizations (as in the proof of the previous proposition), the divergence between two log-normal distributions is equal to the divergence between the corresponding normal distributions. Therefore, using the known formula for the KL divergence of normals, we get the desired result. □

Substituting the expression for the KL divergence in eq. (2.4) inside eq. (2.2), and ignoring for simplicity the special case $f(x) = 0$, we obtain the following loss function for ReLU activations

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x}_i)}[\log p(\mathbf{y}_i|\mathbf{z})] + \beta \log \alpha_\theta(\mathbf{x}_i), \tag{2.6}$$

and a similar expression for Softplus. Notice that the first expectation can be approximated by sampling (in the experiments we use one single sample, as customary for dropout), and is just the average cross-entropy term that is typical in deep learning. The second term, which is new, penalizes the network for choosing a low variance for the noise, i.e. for letting more information pass through to the next layer. This loss can be optimized easily using stochastic gradient descent and the reparametrization trick of Kingma and Welling (2014) to back-propagate the gradient through the sampling operation.

2.7 Connections with Variational Auto-Encoders and ICA

We now outline some important connections between the Information Dropout method, which we just introduced, Variational Autoencoders (VAEs) (Kingma and Welling, 2014), and Independent Component Analysis (ICA) (Comon, 1994; Bell and Sejnow, 1995).

Variational autoencoders aim to reconstruct, given a training dataset $\mathcal{D} = \{\mathbf{x}_i\}$, a latent random variable $\mathbf{z}$ such that the observed data $\mathbf{x}$ can be thought as being generated by the, usually simpler, variable $\mathbf{z}$ through some unknown generative process $p_\theta(\mathbf{x}|\mathbf{z})$. In practice, this is done by minimizing the negative variational lower-bound to the marginal log-likelihood of the data

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x}_i)}[-\log p_\theta(\mathbf{x}_i|\mathbf{z})] + \mathrm{KL}(p_\theta(\mathbf{z}|\mathbf{x}_i) \parallel \prod_i q_\theta(z_i)),$$

where the optimization is joint over the factorized prior $q_\theta(\mathbf{z})$, which is often assumed to be factorized, and the posterior $p_\theta(\mathbf{z}|\mathbf{x})$. The optimization can now be performed easily through sampling using the SGVB method of Kingma and Welling (2014).

We now show that this procedure can be seen as a special case of Information Dropout: Consider again the loss in eq. (2.3) in the special case $\mathbf{y} = \mathbf{x}$, that is, when the task is reconstruction of the input:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{z\sim p(\mathbf{z}|\mathbf{x}_i)}[-\log p(\mathbf{x}_i|\mathbf{z})] + \beta\,\mathrm{KL}(p_\theta(\mathbf{z}|\mathbf{x}_i) \parallel p_\theta(\mathbf{z})) + \gamma\,\mathrm{TC}(\mathbf{z}). \tag{2.7}$$

By Section 2.5, in the special case $\beta = \gamma$, this reduces to

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N} \mathbb{E}_{\mathbf{z}\sim p_\theta(\mathbf{z}|\mathbf{x}_i)}[-\log p_\theta(\mathbf{x}_i|\mathbf{z})] + \beta\,\mathrm{KL}(p_\theta(\mathbf{z}|\mathbf{x}_i) \parallel \prod_i q_\theta(z_i)), \tag{2.8}$$

where again the optimization is joint over prior $q_\theta(\mathbf{z})$ and posterior $p_\theta(\mathbf{z}|\mathbf{x})$, leading to the same optimization problem of a VAE when $\beta = 1$, that is when all quantities have the same weight in the loss function. This derivation also provides some additional insights: when using a factorized prior, a VAE will try to find a representation of the data which is sufficient for reconstruction (cross-entropy term), maximally compressed (KL term) *and* disentangled (total correlation term). We can also see that, while using instead a non factorized prior

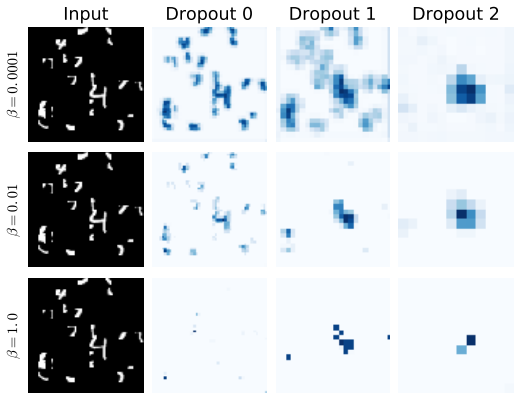


Figure 2.2: Plot of the total KL-divergence at each spatial location in the first three Information Dropout layers (of sizes 48x48, 24x24 and 12x12 respectively) of All-CNN-96 (Springenberg et al., 2014) trained on Cluttered MNIST with different values of β. This measures how much information from each part of the image the Information Dropout layer is transmitting to the next layer. For small β information about the nuisances is transmitted to the next layers, while for higher values of β the dropout layers drop the information as soon as the receptive field is big enough to recognize it as a nuisance. The resulting representation is thus more robust to nuisances, improving generalization. Notice that the noise added by Information Dropout is tailored to the specific sample, to the point that the digit can be localized from the noise mask.

increases the complexity of the optimization problem, it spares the VAE from having to find a disentangled representation, allowing it to obtain a better compression result Kingma et al. (2016). In the same setting as eq. (2.8) we can use larger values of β to force Information Dropout, and hence, in the case of reconstruction, a VAE, to recover representations that are increasingly more compressed and also disentangled. This fact is implicitly used by Higgins et al. (2017b), that derive the loss in eq. (2.8) taking inspiration from experimental evidence in neuroscience. They empirically verify that, as expected from this theoretical derivation, for higher values of β the representation $\mathbf{z}$ recovered by the VAE is increasingly more disentangled.

Equation (2.7) has two other important cases: As we have already seen, the case $\gamma = 0$ and $\beta > 0$ is the standard Information Bottleneck Lagrangian: A VAE trained with this loss will focus purely on compression of the input, without squandering resources to also

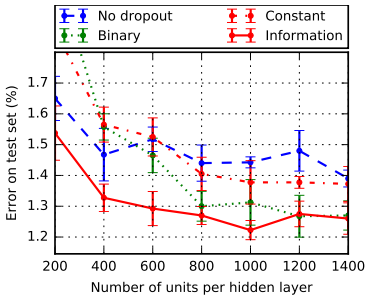
disentangle the representation. In the case $\beta = 0$ and $\gamma > 0$ we obtain instead the standard loss function of Independent Component Analysis (ICA), whereby we try to reconstruct a perfectly disentangled representation of the data, without any constraint on its complexity (quantity of information). While both cases are important on their own right, Section 2.5 does not apply for them, thus the loss function does not generally simplify and cannot be computed in closed form.

2.8 Experiments

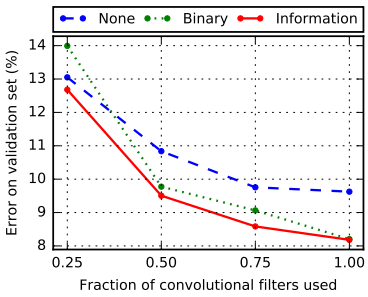
The goal of the experiments in this chapter is to validate the theory, by showing that indeed increasing noise level yields reduced dependency on nuisance factors, a more disentangled representation, and that by adapting the noise level to the data we can better exploit architectures of limited capacity.

To this end, we first compare Information Dropout with the Dropout baseline on several standard benchmark datasets using different networks architecture, and highlight a few key properties. As Kingma et al. (2015) also notice, letting the variance of the noise grow excessively leads to poor generalization. To avoid this problem, we constraint $\alpha(x) < 0.7$, so that the maximum variance of the log-normal error distribution will be approximately 1, the same as binary dropout when using a drop probability of 0.5. In all experiments we divide the KL-divergence term by the number of training samples, so that for $\beta = 1$ the scaling of the KL-divergence term in similar to the one used by Variational Dropout (see Section 2.10).

Cluttered MNIST. To visually asses the ability of Information Dropout to create a representation that is increasingly insensitive to nuisance factors, we train an All-CNN network (Springenberg et al., 2014) for classification on a Cluttered MNIST dataset (Mnih et al., 2014), consisting of 96×96 images containing a single MNIST digit together with 21 distractors. The dataset is divided in 50,000 training images and 10,000 testing images. As shown in Figure 4.5, for small values of β, the network lets through both the objects of interest (digits) and distractors, to upper layers. By increasing the value of β, we force the network to disregard the least discriminative components of the data, thereby building



(a) MNIST



(b) CIFAR-10

Figure 2.3: (a) Average classification error on MNIST over 3 runs of several dropout methods applied to a fully connected network with three hidden layers and ReLU activations. Information dropout outperforms binary dropout, especially on smaller networks, possibly because dropout severely reduces the already limited capacity of the network, while Information Dropout can adapt the amount of noise to the data and the size of the network. Information dropout also outperforms a dropout layer that uses constant log-normal noise with the same variance, confirming the benefits of adaptive noise. (b) Classification error on CIFAR-10 for several dropout methods applied to the All-CNN architecture using Softplus activations.

a better representation for the task. This behavior depends on the ability of Information Dropout to learn the structure of the nuisances in the dataset which, unlike other methods, is facilitated by the ability to select noise level on a per-sample basis.

Occluded CIFAR. Occlusions are a fundamental phenomenon in vision, for which it is difficult to hand-design invariant representations. To assess that the approximate minimal sufficient representation produced by Information Dropout has this invariance property, we created a new dataset by occluding images from CIFAR-10 with digits from MNIST (Figure 2.4). We train the All-CNN network to classify the CIFAR image. The information relative to the occluding MNIST digit is then a nuisance for the task, and therefore should be excluded from the final representation. To test this, we train a secondary network to classify the nuisance MNIST digit using only the the representation learned for the main task. When training with small values of β, the network has very little pressure to limit the effect of nuisances in the representation, so we expect the nuisance classifier to perform better. On the other hand, increasing the value of β we expect its performance to degrade, since the

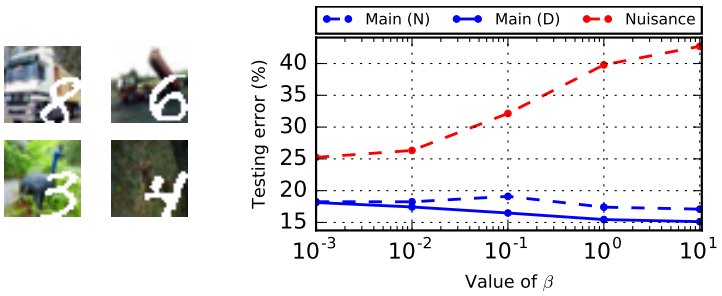


Figure 2.4: **(Left)** Samples from our Occluded CIFAR dataset. **(Right)** Plot of the testing error on the main task (classifying the CIFAR image) and on the nuisance task (classifying the occluding MNIST digit) as β varies. For both tasks, we use the same representation of the data trained for the main task using Information Dropout. For larger values of β the representation is increasingly more invariant to nuisances, making the nuisance classification task harder, but improving the performance on the main task by preventing overfitting. For the nuisance task, we test using the learned noisy representation of the data, since we are interested specifically in the effects of the noise. For the main task, we show the result both using the noisy representation (N), and the deterministic representation (D) obtained by disabling the noise at test time.

representation will become increasingly minimal, and therefore invariant to nuisances. The results in Figure 2.4 confirm this intuition.

MNIST and CIFAR-10. Similar to Kingma et al. (2015), to see the effect of Information Dropout on different network sizes and architectures, we train on MNIST a network with 3 fully connected hidden layers with a variable number of hidden units, and we train on CIFAR-10 (Krizhevsky and Hinton, 2009) an All-CNN network (Springenberg et al., 2014), using a variable percentage of all the filters. The fully connected network was trained for 80 epochs, using stochastic gradient descent with momentum with initial learning rate 0.07 and dropping the learning rate by 0.1 at 30 and 70 epochs. The CNN was trained for 160 epochs with initial learning rate 0.1 and dropping the learning rate by 0.1 at 80 and 120 epochs. We show the results in Figure 2.3. Information Dropout is comparable or outperforms binary dropout, especially on smaller networks. A possible explanation is that dropout severely reduces the already limited capacity of the network, while Information Dropout can adapt the amount of noise to the data and to the size of the network so that the relevant information can still flow to the successive layers. Figure 6.5 shows how the amount of transmitted information adapts to the size and hierarchical level of the layer.

Disentangling. As we saw Section 2.6, in the case of Softplus activations, the logarithm of the activations approximately follow a normal distribution. We can then approximate the total correlation using the associated covariance matrix Σ. Precisely, we have $\mathrm{TC}(\mathbf{z}) = -\log|\Sigma_0^{-1}\Sigma|$, where $\Sigma_0 = \operatorname{diag}\Sigma$ is the variance of the marginal distribution. In Figure 2.5 we plot the testing error and the total correlation of the representation learned by All-CNN-32 on CIFAR-10 when using 25% of the filters for different values of β. As predicted, when β increases the total correlation diminishes, that is, the representation becomes disentangled, and the testing error improves, since we prevent overfitting. When β is to large, information flow is insufficient, and the testing error rapidly increases.

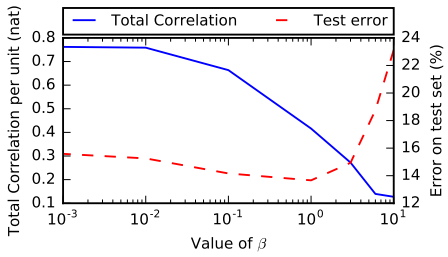


Figure 2.5: For different values of β, plot of the test error and total correlation of the final layer of the All-CNN-32 network with Softplus activations trained on CIFAR-10 with 25% of the filters. Increasing β the test error decreases (we prevent overfitting) and the representation becomes increasingly disentangled. When β is too large, it prevents information from passing through, jeopardizing sufficiency and causing a drastic increase in error.

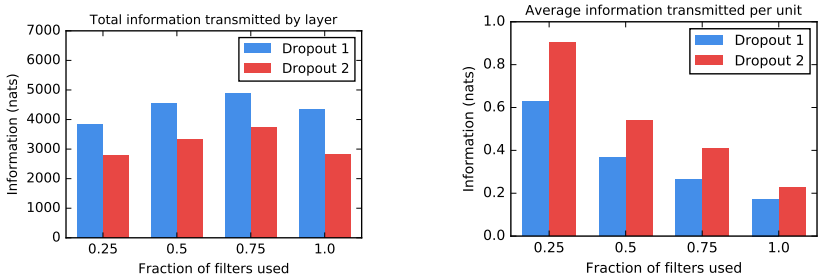


Figure 2.6: Plots of **(Left)** the total information transmitted through the two dropout layers of a All-CNN-32 network with Softplus activations trained on CIFAR and **(Right)** the average quantity of information transmitted through each unit in the two layers. From **(Left)** we see that the total quantity of information transmitted does not vary much with the number of filters and that, as expected, the second layer transmits less information than the first layer, since prior to it more nuisances have been disentangled and discarded. In **(Right)** we see that when we decrease the number of filters, we force each single unit to let more information flow (i.e. we apply less noise), and that the units in the top dropout layer contain on average more information relevant to the task than the units in the bottom dropout layer.

2.9 Discussion

We relate the Information Bottleneck principle and its associated Lagrangian to seemingly unrelated practices and concepts in deep learning, including dropout, disentanglement, variational autoencoding. For classification tasks, we show how an optimal representation can be achieved by injecting multiplicative noise in the activation functions, and therefore into the gradient computation during learning.

A special case of noise (Bernoulli) results in dropout, which is standard practice originally motivated by ensemble averaging rather than information-theoretic considerations. Better (adaptive) noise models result better exploitation of limited capacity, leading to a method we call Information Dropout. We also establish connections with variational inference and variational autoencoding, and show that "disentangling of the hidden causes" can be measured by total correlation and achieved simply by enforcing independence of the components in the representation prior.

So, what may be done by necessity in some computational systems (noisy computation), turns out to be beneficial towards achieving invariance and minimality. Analogously, what has been done for convenience (assuming a factorized prior) turns out to be the beneficial towards achieving "disentanglement."

Another interpretation of Information Dropout is as a way of biasing the network towards reconstructing representations of the data that are compatible with a Markov chain generative model, making it more suited to data coming from hierarchical models, and in this sense is complementary to architectural constraint, such as convolutions, that instead bias the model toward geometric tasks.

It should be noticed that injecting multiplicative noise to the activations can be thought of as a particular choice of a class of minimizers of the loss function, but can also be interpreted as a regularization terms added to the cost function, or as a particular procedure utilized to carry out the optimization. So the same operation can be interpreted as either of the three key ingredients in the optimization: the function to be minimized, the family over which to minimize, and the procedure with which to minimize. This highlight the intimate interplay

between the choice of models and algorithms in deep learning.

2.10 Related work

The main contribution of this chapter is to establish how two seemingly different areas of research, namely dropout methods to prevent overfitting, and the study of optimal representations, can be linked through the Information Bottleneck principle.

Dropout was introduced by Srivastava et al. (2014). The original motivation was that by randomly dropping the activations during training, we can effectively train an ensemble of exponentially many networks, that are then averaged during testing, therefore reducing overfitting. Wang and Manning (2013) suggested that dropout could be seen as performing a Monte-Carlo approximation of an implicit loss function, and that instead of multiplying the activations by binary noise, like in the original dropout, multiplicative Gaussian noise with mean 1 can be used as a way of better approximating the implicit loss function. This led to a comparable performance but faster training than binary dropout.

The interpretation of deep neural network as a way of creating successively better representations of the data has already been suggested and explored by many. Most recently, Tishby and Zaslavsky (2015) put forth an interpretation of deep neural networks as creating sufficient representations of the data that are increasingly minimal. In parallel simultaneous work, Alemi et al. (2016) approximate the information bottleneck similarly to us, but focus on empirical analysis of robustness to adversarial perturbations rather than tackling disentanglement, invariance and minimality analytically.

Sufficient dimensionality reduction (Adragni and Cook, 2009) and Optimal Component Analysis (Liu et al., 2003) follow a similar idea to us, in that they focus on finding the smallest (usually linear) sufficient statistic of the data that is sufficient for a given task. However, while they define small in term of dimension of the representation, we focus on finding a (non-linear) representation with minimal information content, but whose dimension can, in fact, be even larger than the original data. By allowing large non-linear representations, we can exploit the full representational power of deep networks, while the minimality of

the information content still promotes nuisance invariance and prevents overfitting. Our framework also has connections with Independent Component Analysis (ICA).

Some have focused on creating representations that are *maximally* invariant to nuisances, especially when they have the structure of a (possibly infinite-dimensional) group acting on the data, like Sundaramoorthi et al. (2009), or, when the nuisance is a locally compact group acting on each layer, by successive approximations implemented by hierarchical convolutional architectures, like Anselmi et al. (2016) and Bruna and Mallat (2011). In these cases, which cover common nuisances such as translations and rotations of an image (affine group), or small diffeomorphic deformations due to a slight change of point of view (group of diffeomorphisms), the representation is equivalent to the data modulo the action of the group. However, when the nuisances are not a group, as is the case for occlusions, it is not possible to achieve such equivalence, that is, there is a loss. To address this problem, Soatto and Chiuso (2016) defined optimal representations not in terms of maximality, but in terms of *sufficiency*, and characterized representations that are both sufficient and invariant. They argue that the management of nuisance factors common in visual data, such as changes of viewpoint, local deformations, and changes of illumination, is directly tied to the specific structure of deep convolutional networks, where local marginalization of simple nuisances at each layer results in marginalization of complex nuisances in the network as a whole.

Our work fits in this last line of thinking, where the goal is not equivalence to the data up to the action of (group) nuisances, but instead sufficiency for the task. Our main contribution in this sense is to show that injecting noise into the layers, and therefore using a non-deterministic function of the data, can actually simplify the theoretical analysis and lead to disentangling and improved insensitivity to nuisances. This is an alternate explanation to that put forth by the references above.

CHAPTER 3

Disentangled and compositional representations

A critical feature of biological intelligence is its capacity for *life-long learning* (Cichon and Gan, 2015) – the ability to acquire new knowledge from a sequence of experiences to solve progressively more tasks, while maintaining performance on previous ones. This, however, remains a serious challenge for current deep learning approaches. While current methods are able to outperform humans on many individual problems (Silver et al., 2016; Mnih et al., 2015; He et al., 2015), these algorithms suffer from *catastrophic forgetting*: Training on a new task or environment can be enough to degrade their performance from super-human to chance level (Rusu et al., 2016). Another critical aspect of life-long learning is the ability to sensibly reuse previously learnt representations in new domains (*positive transfer*). For example, knowing that strawberries and bananas are not edible when they are green could be useful when deciding whether to eat a green peach in the future. Finding semantic homologies between visually distinctive domains can remove the need to learn from scratch on every new environment and hence help with data efficiency – another major drawback of current deep learning approaches (Garnelo et al., 2016; Lake et al., 2016).

In this chapter we show how a disentangled representations, which we discussed in Chapter 2, can be used in a life-long learning, exploiting this setting to learn shared representations across domains where applicable. The proposed Variational Autoencoder with Shared Embeddings (VASE, see fig. 3.1B) automatically detects shifts in the training data distribution and uses this information to allocate spare latent capacity to novel dataset-specific disentangled representations, while reusing previously acquired representations of latent dimensions where applicable. We use latent masking and a generative "dreaming" feedback loop (similar to Ramapuram et al. (2017); Shin et al. (2017); Seff et al. (2017); Ans and Rousset

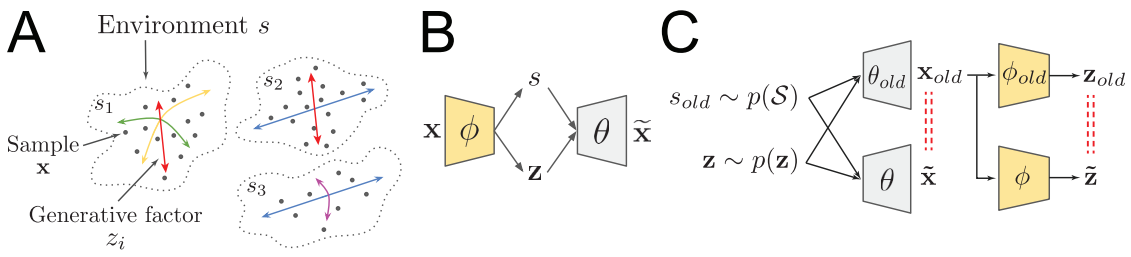


Figure 3.1: **A**: Schematic representation of the life-long learning data distribution. Each dataset/environment corresponds to a cluster s. Data samples $\mathbf{x}$ constituting each cluster can be described by a local set of coordinates (data generative factors z_n). Different clusters may share some data generative factors. **B**: VASE model architecture **C**: ConContinSchematic of the "dreaming" feedback loop. We use a snapshot of the model with the old parameters (ϕ_{old}, θ_{old}) to generate an imaginary batch of data $\mathbf{x}_{\text{old}}$ for a previously experienced dataset s_{old}. While learning in the current environment, we ensure that the representation is still consistent on the hallucinated "dream" data, and can reconstruct it (see red dashed lines).

(1997)) to avoid catastrophic forgetting. Furthermore, we demonstrate that the pressure to disentangle endows VASE with a number of useful properties: 1) dealing sensibly with ambiguous inputs; 2) learning richer representations through imagination-based exploration; 3) performing semantically meaningful cross-domain inference by ignoring irrelevant aspects of surface-level appearance.

3.1 Problem formalization and implementation

Problem formalisation

We assume that there is an a priori unknown set $\mathcal{S} = \{s_1, s_2, ..., s_K\}$ of K environments which, between them, share a set $\mathcal{Z} = \{z_1, z_2, ..., z_N\}$ of N independent data generative factors. We assume $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. Since we aim to model piece-wise stationary data, it is reasonable to assume $s \sim \text{Cat}(\pi_{1,...,K})$, where π_k is the probability of observing environment s_k. Two environments may use the same generative factors but render them differently, or they may use a different subset of factors altogether. Given an environment s, and an

environment-dependent subset $\mathcal{Z}^s \subseteq \mathcal{Z}$ of the ground truth generative factors, it is possible to synthesise a dataset of images $\mathbf{x}^s \sim p(\cdot|\mathbf{z}^s, s)$. In order to keep track of which subset of the N data generative factors is used by each environment s to generate images $\mathbf{x}^s$, we introduce an environment-dependent mask $\mathbf{a}^s$ with dimensionality $|\mathbf{a}| = N$, where $a_n^s = 1$ if $z_n \in \mathcal{Z}^s$ and zero otherwise. Hence, we assume $\mathbf{a}^s \sim \text{Bern}(\omega^s_{1,\dots,N})$, where ω_n^s is the probability that factor z_n is used in environment s. This leads to the following generative process (where "$\odot$" is element-wise multiplication):

$$\begin{gathered} \mathbf{z} \sim \mathcal{N}(\mathbf{0},\ \mathbf{I}), \qquad s \sim \text{Cat}(\pi_{1,\dots,K}), \qquad \mathbf{a}^s \sim \text{Bern}(\omega^s_{1,\dots,N}), \\ \mathbf{z}^s = \mathbf{a}^s \odot \mathbf{z}, \qquad \mathbf{x}^s \sim p(\cdot \mid \mathbf{z}^s, s) \end{gathered} \tag{3.1}$$

Intuitively, we assume that the piece-wise stationary observed data $\mathbf{x}$ can be split into *clusters* (environments s) (note evidence for similar experience clustering from the animal literature (Auchter et al., 2017)). Each cluster has a set of *standard coordinate axes* (a subset of the generative factors $\mathbf{z}$ chosen by the latent mask $\mathbf{a}^s$) that can be used to parametrise the data in that cluster (fig. 3.1A). Given a sequence $\mathbf{x} = (\mathbf{x}^{s_1},\ \mathbf{x}^{s_2},\ \dots)$ of datasets generated according to the process in eq. (3.1), where $s_k \sim p(\mathbf{s})$ is the k-th sample of the environment, the aim of life-long representation learning can be seen as estimating the full set of generative factors $\mathcal{Z} \approx \bigcup_k q(\mathbf{z}^{s_k}|\mathbf{x}^{s_k})$ from the environment-specific subsets of $\mathbf{z}$ inferred on each stationary data cluster $\mathbf{x}^{s_k}$. Henceforth, we will drop the subscript k for simplicity of notation.

Inferring the data generative factors

Observations $\mathbf{x}^s$ cannot contain information about the generative factors z_n that are not relevant for the environment s. Hence, we use the following form for representing the data generative factors:

$$q(\mathbf{z}^s|\mathbf{x}^s) = \mathbf{a}^s \ \odot\ \mathcal{N}(\mu(\mathbf{x}),\ \sigma(\mathbf{x})) + (1 - \mathbf{a}^s)\ \odot\ \mathcal{N}(\mathbf{0},\ \mathbf{I}). \tag{3.2}$$

Note that μ and σ in eq. (3.2) depend only on the data $\mathbf{x}$ and not on the environment s. This is important to ensure that the semantic meaning of each latent dimension z_n remains

consistent for different environments s. We model the representation $q(\mathbf{z}^s|\mathbf{x}^s)$ of the data generative factors as a product of independent normal distributions to match the assumed prior $p(\mathbf{z}) \sim \mathcal{N}(\mathbf{0},\ \mathbf{I})$.

In order to encourage the representation $q(\mathbf{z}^s|\mathbf{x}^s)$ to be semantically meaningful, we encourage it to capture the generative factors of variation within the data $\mathbf{x}^s$ by following the MDL principle. We aim to find a representation $\mathbf{z}^s$ that minimises the reconstruction error of the input data $\mathbf{x}^s$ conditioned on $\mathbf{z}^s$ under a constraint on the quantity of information in $\mathbf{z}^s$. This leads to the following loss function:

$$\mathcal{L}_{\text{MDL}}(\phi, \theta) = \underbrace{\mathbb{E}_{\mathbf{z}^s \sim q_\phi(\cdot|\mathbf{x}^s)}[-\log\ p_\theta(\mathbf{x} \mid \mathbf{z}^s, s)]}_{\text{Reconstruction error}} + \gamma\, |\, \underbrace{\mathbb{KL}\left(q_\phi(\mathbf{z}^s|\mathbf{x}^s)||p(\mathbf{z})\right)}_{\text{Representation capacity}} - \underbrace{C}_{\text{Target}}\, |^2 \tag{3.3}$$

The loss in eq. (3.3) is closely related to the β-VAE (Higgins et al., 2017a) objective $\mathcal{L} = \mathbb{E}_{\mathbf{z} \sim q_\phi(\cdot|\mathbf{x})}[-\log\ p_\theta(\mathbf{x}|\mathbf{z})] + \beta\ \mathbb{KL}\left(q_\phi(\mathbf{z}|\mathbf{x})||p(\mathbf{z})\right)$, which uses a Lagrangian to limit the latent bottleneck capacity, rather than an explicit target C. It was shown that optimising the β-VAE objective helps with learning a more semantically meaningful disentangled representation $q(\mathbf{z}|\mathbf{x})$ of the data generative factors (Higgins et al., 2017a). However, Burgess et al. (2017) showed that progressively increasing the target capacity C in eq. (3.3) throughout training further improves the disentanglement results reported by Higgins et al. (2017a), while simultaneously producing sharper reconstructions. Progressive increase of the representational capacity also seems intuitively better suited to continual learning where new information is introduced in a sequential manner. Hence, VASE optimises the objective function in eq. (3.3) over a sequence of datasets $\mathbf{x}^s$. This, however, requires a way to infer s and $\mathbf{a}^s$, as discussed next.

Inferring the latent mask

Given a dataset $\mathbf{x}^s$, we want to infer which latent dimensions z_n were used in its generative process (see eq. (3.1)). This serves multiple purposes: 1) helps identify the environment s (see next section); 2) helps ignore latent factors z_n that encode useful information in some environment but are not used in the current environment s, in order to prevent retraining and subsequent catastrophic forgetting; and 3) promotes latent sharing between environments.

Remember that eq. (3.3) indirectly optimises for $\mathbb{E}_{\mathbf{x}^s}[q_\phi(\mathbf{z}^s|\mathbf{x}^s)] \approx p(\mathbf{z})$ after training on a dataset s. If a new dataset uses the same generative factors as $\mathbf{x}^s$, then the marginal behaviour of the corresponding latent dimensions z_n will not change. On the other hand, if a latent dimension encodes a data generative factor that is irrelevant to the new dataset, then it will start behaving atypically and stray away from the prior. We capture this intuition by defining the *atypicality* score α_n for each latent dimension z_n on a batch of data $\mathbf{x}^s_{\text{batch}}$:

$$\alpha_n = \mathbb{KL}\left(\mathbb{E}_{\mathbf{x}^s_{\text{batch}}}[\ q_\phi(z^s_n|\mathbf{x}^s_{\text{batch}})\ ]\ ||\ p(z_n)\right). \tag{3.4}$$

The atypical components are unlikely to be relevant to the current environment, so we mask them out:

$$a^s_n = \begin{cases} 1, \text{ if } \alpha_n < \lambda \\ \\ 0, \text{ otherwise} \end{cases} \tag{3.5}$$

where λ is a threshold hyperparameter (see supplementary material of Achille et al. (2018) for details). Note that the uninformative latent dimensions z_n that have not yet learnt to represent any data generative factors, i.e. $q_\phi(z_n|\mathbf{x}^s_n) = p(z_n)$, are automatically unmasked in this setup. This allows them to be available as spare latent capacity to learn new generative factors when exposed to a new dataset. Fig. 3.2 shows the sharp changes in α_n at dataset boundaries during training.

Inferring the environment

Given the generative process introduced in eq. (3.1), it may be tempting to treat the environment s as a discrete latent variable and learn it through amortised variational inference. However, we found that in the continual learning scenario this is not a viable strategy. Parametric learning is slow, yet we have to infer each new data cluster s extremely fast to avoid catastrophic forgetting. Hence, we opt for a fast non-parametric meta-algorithm motivated by the following intuition. Having already experienced r datasets during life-long learning, there are two choices when it comes to inferring the current one s: it is either a new dataset s_{r+1}, or it is one of the r datasets encountered in the past. Intuitively, one way to check for the former is to see whether the current data $\mathbf{x}^s$ seems likely under any of the previously

seen environments. This condition on its own is not sufficient though. It is possible that environment s uses a subset of the generative factors used by another environment $\mathcal{Z}^s \subseteq \mathcal{Z}^t$, in which case environment t will explain the data $\mathbf{x}^s$ well, yet it will be an incorrect inference. Hence, we have to ensure that the subset of the relevant generative factors $\mathbf{z}^s$ inferred for the current data $\mathbf{x}^s$ according to section 3.1 matches that of the candidate past dataset t. Given a batch $\mathbf{x}^s_{\text{batch}}$, we infer the environment s according to:

$$s = \begin{cases} \hat{s} & , \quad \text{if} \;\; \mathbb{E}_{\mathbf{z}^{\hat{s}}}[\; p_\theta(\mathbf{x}^s_{\text{batch}}|\mathbf{z}^{\hat{s}}, \hat{s})\;] \;\leq \kappa L_{\hat{s}} \quad \wedge \quad \mathbf{a}^s = \mathbf{a}^{\hat{s}} \\ s_{r+1}, & \text{otherwise} \end{cases} \tag{3.6}$$

where $\hat{s} = \arg\max_s \; q(s|\mathbf{x}^s_{\text{batch}})$ is the output of an auxiliary classifier trained to infer the most likely previously experienced environment $\hat{s}$ given the current batch $\mathbf{x}^s_{\text{batch}}$, $L_{\hat{s}}$ is the average reconstruction error observed for the environment $\hat{s}$ when it was last experienced, and κ is a threshold hyperparameter (see supplementary material of Achille et al. (2018) for details).

Preventing catastrophic forgetting

So far we have discussed how VASE integrates knowledge from the current environment into its representation $q_\phi(\mathbf{z}|\mathbf{x})$, but we have not yet discussed how we ensure that past knowledge is not forgotten in the process. Most standard approaches to preventing catastrophic forgetting discussed in Section 3.4 are either not applicable to a variational context, or do not scale well due to memory requirements. However, thanks to learning a generative model of the observed environments, we can prevent catastrophic forgetting by periodically *hallucinating* (i.e. generating samples) from past environments using a snapshot of VASE, and making sure that the current version of VASE is still able to model these samples. A similar "dreaming" feedback loop was used by Ramapuram et al. (2017); Shin et al. (2017); Seff et al. (2017); Ans and Rousset (1997).

More formally, we follow the generative process in eq. (3.1) to create a batch of samples $\mathbf{x}_{\text{old}} \sim q_{\theta_{\text{old}}}(\cdot|\mathbf{z}, s_{\text{old}})$ using a snapshot of VASE with parameters $(\phi_{\text{old}},\ \theta_{\text{old}})$ (see fig. 3.1C).

We then update the current version of VASE according to the following (replacing $_{\text{old}}$ with $'$ for brevity):

$$\mathcal{L}_{\text{past}}(\phi,\theta) = \mathbb{E}_{\mathbf{z},s',\mathbf{x}'}\Big[\underbrace{D[q_\phi(\mathbf{z}|\mathbf{x}'),\ q_{\phi'}(\mathbf{z}'|\mathbf{x}')]}_{\text{Encoder proximity}} + \underbrace{D[q_\theta(\mathbf{x}|\mathbf{z},s'),\ q_{\theta'}(\mathbf{x}'|\mathbf{z},s')]}_{\text{Decoder proximity}} \Big], \tag{3.7}$$

where D is a distance between two distributions (we use the Wasserstein distance for the encoder and KL divergence for the decoder). The snapshot parameters get synced to the current trainable parameters $\phi_{\text{old}} \leftarrow \phi$, $\theta_{\text{old}} \leftarrow \theta$ every τ training steps, where τ is a hyperparameter. The expectation over simulators s_{old} and latents $\mathbf{z}$ in eq. (3.7) is done using Monte Carlo sampling.

Model summary

To summarise, we train our model using a meta-algorithm with both parametric and non-parametric components. The latter is needed to quickly associate new experiences to an appropriate cluster, so that learning can happen inside the current experience cluster, without disrupting unrelated clusters. We initialise the latent representation $\mathbf{z}$ to have at least as many dimensions as the total number of the data generative factors $|\mathbf{z}| \geq |\mathcal{Z}| = N$, and the softmax layer of the auxiliary environment classifier to be at least as large as the number of datasets $|\mathcal{S}| = K$. As we observe the sequence of training data, we detect changes in the environment and dynamically update the internal estimate of $r \leq K$ datasets experienced so far according to eq. (3.6). We then train VASE by minimising the following objective function:

$$\begin{aligned}\mathcal{L}(\phi,\theta) = &\underbrace{\mathbb{E}_{\mathbf{z}^s\sim q_\phi(\cdot|\mathbf{x}^s))}[-\log p_\theta(\mathbf{x}|\mathbf{z}^s,s)] \ + \ \gamma\,|\mathbb{KL}\left(q_\phi(\mathbf{z}^s|\mathbf{x}^s)||p(\mathbf{z})\right) - C|^2}_{\text{MDL on current data}} + \\ &+ \underbrace{\mathbb{E}_{\mathbf{z},s',\mathbf{x}'}\Big[D[q_\phi(\mathbf{z}|\mathbf{x}'),\ q_{\phi'}(\mathbf{z}'|\mathbf{x}')] + D[q_\theta(\mathbf{x}|\mathbf{z},s'),\ q_{\theta'}(\mathbf{x}'|\mathbf{z},s')] \Big]}_{\text{“Dreaming” feedback on past data}}.\end{aligned} \tag{3.8}$$

3.2 Experiments

Continual learning with disentangled shared latents First, we qualitatively assess whether VASE is able to learn good representations in a continual learning setup. We use

a sequence of three datasets: (1) a moving version of Fashion-MNIST (Xiao et al., 2017) (shortened to moving Fashion), (2) MNIST (LeCun et al., 1998), and (3) a moving version of MNIST (moving MNIST). During training we expect VASE to detect shifts in the data distribution and dynamically create new experience clusters s, learn a disentangled representation of each environment without forgetting past environments, and share disentangled factors between environments in a semantically meaningful way. Fig. 3.2 (top) compares the performance of VASE to that of Controlled Capacity Increase-VAE (CCI-VAE) (Burgess et al., 2017), a model for disentangled representation learning with the same architecture as VASE but without the modifications introduced in this paper to allow for continual learning. It can be seen that unlike VASE, CCI-VAE forgot moving Fashion at the end of the training sequence. Both models were able to disentangle position from object identity, however, only VASE was able to meaningfully share latents between the different datasets - the two positional latents are active for two moving datasets but not for the static MNIST. VASE also has moving Fashion- and MNIST-specific latents, while CCI-VAE shares all latents between all datasets. VASE use only $8/24$ latent dimensions at the end of training. The rest remained as spare capacity for learning on future datasets.

Learning representations for tasks We train object identity classifiers (one each for moving Fashion and MNIST) and an object position regressor on top of the latent representation $\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})$ at regular intervals throughout the continual learning sequence. Good accuracy on these measures would indicate that at the point of measurement, the latent representation $\mathbf{z}$ contained dataset relevant information, and hence could be useful, e.g. for subsequent policy learning in RL agents. Figure 3.2 (bottom) shows that both VASE and CCI-VAE learn progressively more informative latent representations when exposed to each dataset s, as evidenced by the increasing classification accuracy and decreasing mean squared error (MSE) measures within each stage of training. However, with CCI-VAE, the accuracy and MSE measures degrade sharply once a domain shift occurs. This is not the case for VASE, which retains a relatively stable representation.

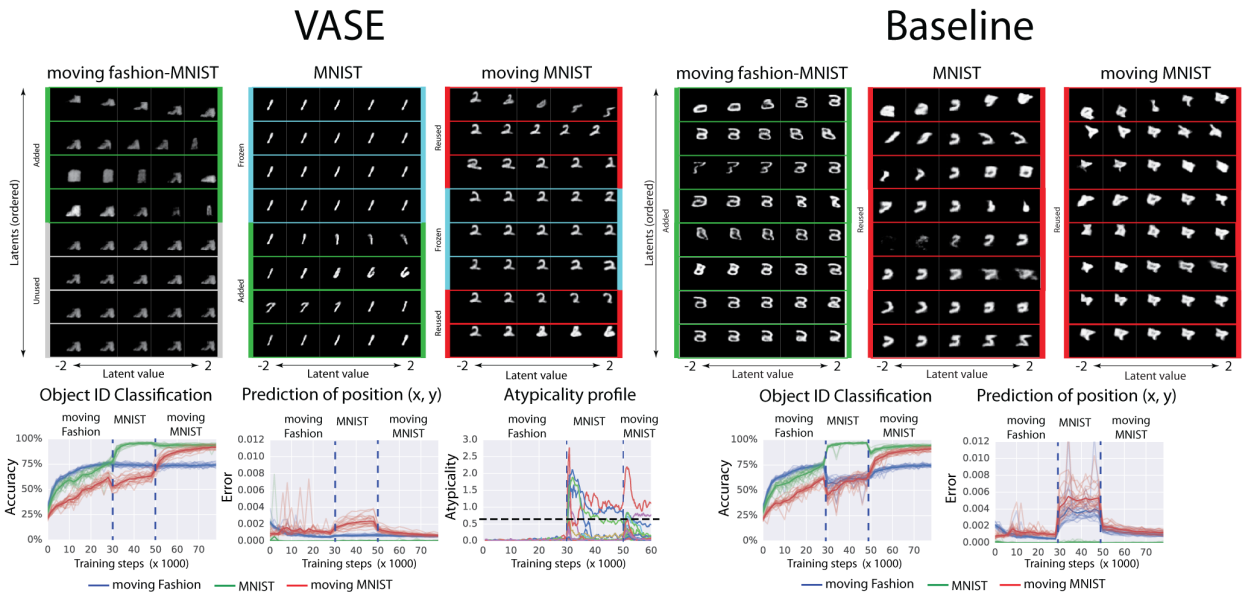


Figure 3.2: We compare VASE to a CCI-VAE baseline. Both are trained on a sequence of three datasets: moving fashion MNIST (moving Fashion) → MNIST → moving MNIST. **Top**: latent traversals at the end of training seeded with samples from the three datasets. The value of each latent z_n is traversed between -2 and 2 one at a time, and the corresponding reconstructions are shown. Rows correspond to latent dimensions z_n, columns correspond to the traversal values. Latent use progression throughout training is demonstrated in colour. **Bottom**: performance of MNIST and Fashion object classifiers and a position regressor trained on the latent space $\mathbf{z}$ throughout training. Note the relative stability of the curves for VASE compared to the baseline. The atypicality profile shows the values of α_n through training (different colours indicate different latent dimensions), with the threshold λ indicated by the dashed black line.

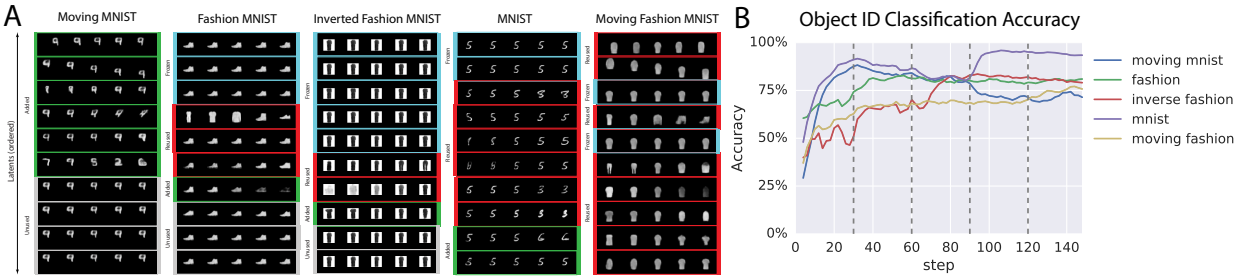


Figure 3.3: Latent traversals (**A**) and classification accuracy (**B**) (both as in fig. 3.2) for VASE trained on a sequence of moving MNIST → Fashion → inverse Fashion → MNIST → moving Fashion. See **??** for larger traversals.

Ablation study Here we perform a full ablation study to test the importance of the proposed components for unsupervised life-long representation learning: 1) regularisation towards disentangled representations (Section 3.1), 2) latent masking (Section 3.1 - **A**), 3) environment clustering (Section 3.1 - **S**), and 4) "dreaming" feedback loop (Section 3.1 - **D**). We use the constraint capacity loss in eq. (3.3) for the disentangled experiments, and the standard VAE loss (Kingma and Welling, 2014; Rezende et al., 2014) for the entangled experiments (Higgins et al., 2017a). For each condition we report the average change in the classification metrics reported above, and the average maximum values achieved (see **??** for details). Table 3.1 shows that the unablated VASE (SDA) has the best performance. Note that the entangled baselines perform worse than the disentangled equivalents, and that the capacity constraint of the CCI-VAE framework does not significantly affect the maximal classification accuracy compared to the VAE. It is also worth noting that VASE outperforms the entangled SD condition, which is similar to the only other baseline VAE-base approach to continual learning that we are aware of Ramapuram et al. (2017). We have also trained VASE on longer sequences of datasets (moving MNIST → Fashion → inverse Fashion → MNIST → moving Fashion) and found similar levels of performance (see fig. 3.3).

Dealing with ambiguity Natural stimuli are often ambiguous and may be interpreted differently based on contextual clues. Examples of such processes are common, e.g. visual illusions like the Necker cube (Necker, 1832), and may be driven by the functional organisa-

	Disentangled				Entangled			
	Object ID Accuracy		Position MSE		Object ID Accuracy		Position MSE	
Ablation	Max (%)	Change (%)	Min (*1e-4)	Change (*1e-4)	Max (%)	Change (%)	Min (*1e-4)	Change (*1e-4)
-	88.6 (±0.4)	-15.2 (±2.8)	3.5 (±0.05)	24.8 (±13.5)	91.8 (±0.4)	-12.1 (±0.8)	4.2 (±0.7)	10.5 (±2.6)
S	88.9 (±0.5)	-13.9 (±1.9)	3.4 (±0.05)	22.5 (±12.2)	91.7 (±0.4)	-12.2 (±0.03)	4.5 (±0.8)	10.9 (±3.1)
D	88.6 (±0.3)	-14.4 (±1.9)	3.3 (±0.04)	21.4 (±4.9)	91.8 (±0.4)	-12.4 (±0.7)	4.3 (±0.7)	11.7 (±3.2)
A	86.7 (±1.9)	-24.5 (±1.0)	3.3 (±0.04)	67.6 (±107.0)	88.6 (±0.3)	-19.7 (±0.5)	4.5 (±0.7)	47.1 (±26.2)
SA	87.1 (±1.8)	-28.1 (±0.08)	3.3 (±0.04)	78.9 (±109.0)	89.9 (±1.3)	-18.3 (±0.4)	4.8 (±0.7)	41.8 (±20.6)
DA	86.3 (±2.5)	-25.2 (±0.5)	3.3 (±0.04)	72.2 (±90.0)	88.8 (±0.3)	-19.4 (±0.4)	4.6 (±0.7)	40.2 (±19.2)
SD	88.3 (±0.3)	-12.9 (±1.9)	3.4 (±0.05)	20.0 (±3.5)	91.4 (±0.3)	-11.7 (±0.6)	4.3 (±0.5)	11.6 (±1.9)
VASE (SDA)	88.6 (±0.4)	**-5.4 (±0.3)**	3.2 (±0.03)	**3.0 (±0.2)**	91.5 (±0.1)	-6.5 (±0.7)	4.2 (±0.4)	3.9 (±1.1)

Table 3.1: Average change in classification accuracy/MSE and maximum/minimum average accuracy/MSE when training an object/position classifier/regressor on top of the learnt representation on the moving Fashion → MNIST → moving MNIST sequence. We do a full ablation study of VASE, where D - dreaming feedback loop, S - cluster inference $q(s|\mathbf{x}^s)$, and A - atypicality based latent mask $\mathbf{a}^s$ inference. We compare two versions of our model - one that is encouraged to learn a disentangled representation through the capacity increase regularisation in eq. (3.3), and an entangled VAE baseline ($\beta = 1$). The unablated disentangled version of VASE (SDA) has the best performance.

tion and the heavy top-down influences within the ventral visual stream of the brain (Gulyas et al., 1993; Przybyszewski, 1998). To evaluate the ability of VASE to deal with ambiguous inputs based on the context, we train it on a CelebA (Ziwei Liu, 2015) → inverse Fashion sequence, and test it using ambiguous linear interpolations between samples from the two datasets (fig. 3.4A, first row). To measure the effects of ambiguity, we varied the interpolation weights between the two datasets. To measure the effects of context, we presented the ambiguous samples in a batch with real samples from one of the training datasets, varying the relative proportions of the two. Figure 3.4A (bottom) shows the inferred probability of interpreting the ambiguous samples as CelebA $q_\phi(s = \text{celebA}|\mathbf{x})$. VASE shows a sharp boundary between interpreting input samples as Fashion or CelebA despite smooth changes in input ambiguity. Such *categorical perception* is also characteristic of biological intelligence (Etcoff and Magee, 1992; Freedman, 2001; Liu and Jagadeesh, 2008). The decision bound-

ary for categorical perception is affected by the context in which the ambiguous samples are presented. VASE also represents its uncertainty about the ambiguous inputs by increasing the inferred variance of the relevant latent dimensions (fig. 3.4A, second row).

Semantic transfer Here we test whether VASE can learn more sophisticated cross-domain latent homologies than the positional latents on the moving MNIST and Fashion datasets described above. Hence, we trained VASE on a sequence of two visually challenging DMLab-30 [1] (Beattie et al., 2016) datasets: the Exploit Deferred Effects (EDE) environment and a randomized version of the Natural Labyrinth (NatLab) environment (Varying Map Randomized). While being visually very distinct (one being indoors and the other outdoors), the two datasets share many data generative factors that have to do with the 3D geometry of the world (e.g. horizon, walls/terrain, objects/cacti) and the agent's movements (first person optic flow). Hence, the two domains share many semantically related factors $\mathbf{z}$, but these are rendered into very different visuals $\mathbf{x}$. We compared cross-domain reconstructions of VASE and an equivalent entangled VAE ($\beta = 1$) baseline. The reconstructions were produced by first inferring a latent representation based on a batch from one domain, e.g. $\mathbf{z}^{\text{NatLab}} = q_\phi(\cdot|\mathbf{x}^{\text{NatLab}})$, and then reconstructing them conditioned on the other domain $\mathbf{x}^{\text{xRec}} = q_\theta(\cdot|\mathbf{z}^{\text{NatLab}}, s^{\text{EDE}})$. Fig. 3.4 shows that VASE discovered the latent homologies between the two domains, while the entangled baseline failed to do so. VASE learnt the semantic equivalence between the cacti in NatLab and the red objects in EDE, the brown fog corresponding to the edge of the NatLab world and the walls in EDE (top leftmost reconstruction), and the horizon lines in both domains. The entangled baseline, on the other hand, seemed to rely on the surface-level pixel statistics and hence struggled to produce meaningful cross-domain reconstructions, attempting to match the texture rather than the semantics of the other domain. See **??** for additional cross-domain reconstructions, including on the sequence of five datasets mentioned earlier.

[1] `https://github.com/deepmind/lab/tree/master/game_scripts/levels/contributed/dmlab30#dmlab-30`

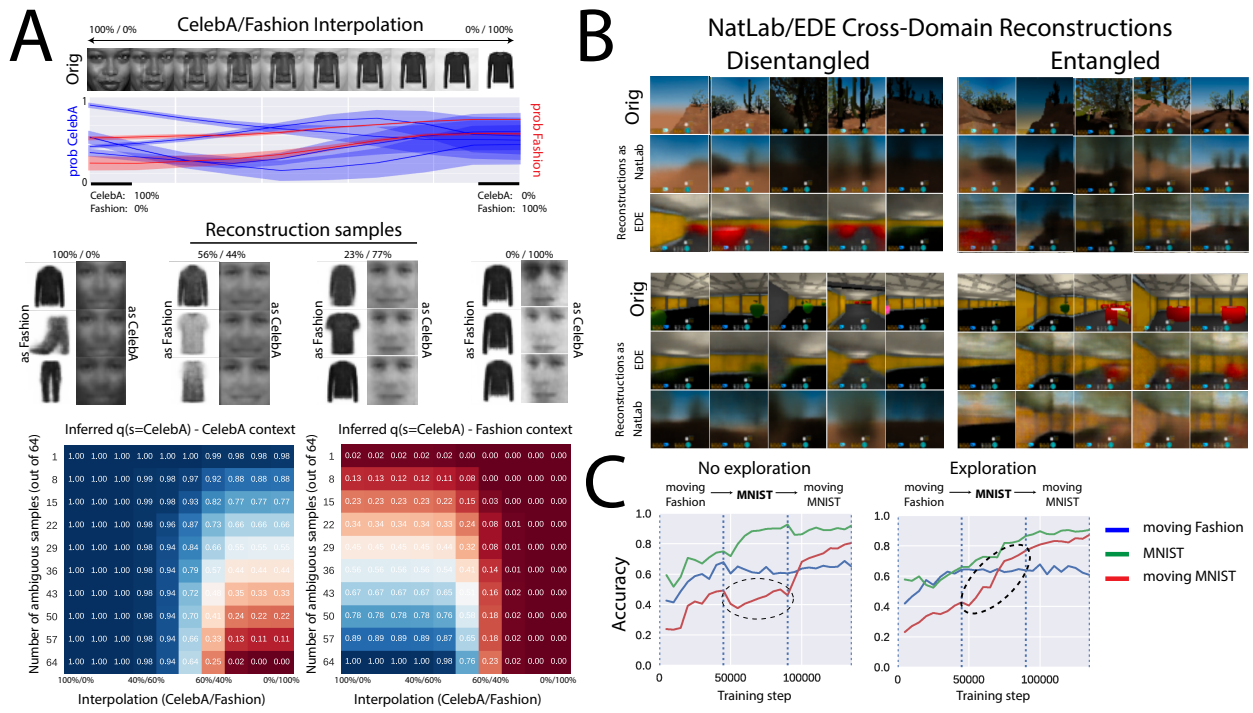


Figure 3.4: **A** Top: Ambiguous input examples created by using different interpolation weights between samples from CelebA and Fashion, and corresponding inferred parameters μ (y axis) and σ (light colour range) of $q_\phi(\mathbf{z}|\mathbf{x})$; red corresponds to Fashion-specific latents, blue to CelebA-specific latents. Middle: Reconstruction samples $p_\theta(\mathbf{x}^s|\mathbf{z}^s, s)$ for different levels of ambiguity conditioned on either dataset. Bottom: Inferred $q_\psi(s = \text{CelebA}$ given different levels of input ambiguity (x axis) and different number of ambiguous vs real data samples (y axis) for the two datasets. VASE deals well with ambiguity, shows context-dependent categorical perception and uncertainty within its inferred representation parameters. **B** Cross-domain reconstructions on NatLab (outdoors) or EDE (indoors) DM Lab levels. The disentangled VASE finds semantic homologies between the two datasets (e.g. cacti $\rightarrow$ red objects). The entangled VASE only maps lower level statistics. **C** Imagination-based exploration allows VASE to imagine the possibility of moving MNIST digits during static MNIST training by using position latents acquired on moving Fashion. This helps it learn a moving MNIST classifier during static MNIST training without ever seeing real translations of MNIST digits.

Imagination-driven exploration Once we learn the concept of moving objects in one environment, it is reasonable to imagine that a novel object encountered in a different environment can also be moved. Given the ability to act, we may try to move the object to realise our hypothesis. We can use such imagination-driven exploration to augment our experiences in an environment and let us learn a richer representation. Notice however, that such imagination requires a compositional representation that allows for novel yet sensible recombinations of previously learnt semantic factors. We now investigate whether VASE can use such imagination-driven exploration to learn better representations using a sequence of three datasets: moving Fashion $\rightarrow$ MNIST $\rightarrow$ moving MNIST. During the first moving Fashion stage, VASE learns the concepts of position and Fashion sprites. It also learns how to move the sprites to reach the imagined states z^* by training an auxiliary policy (see **??** for details). It can then use this policy to do an imagination-based augmentation of the input data on MNIST by imagining MNIST digits in different positions and transforming the static sprites correspondingly using the learnt policy. Hence, VASE can imagine the existence of moving MNIST before actually experiencing it. Indeed, fig. 3.4C shows that when we train a moving MNIST classifier during the static MNIST training stage, the classifier is able to achieve good accuracy in the *imagination-driven exploration* condition, highlighting the benefits of imagination-driven data augmentation.

3.3 Discussion

We have introduced VASE, a novel approach to life-long unsupervised representation learning that builds on recent work on disentangled factor learning (Higgins et al., 2017a; Burgess et al., 2017) by introducing several new key components. Unlike other approaches to continual learning, our algorithm does not require us to maintain a replay buffer of past datasets, or to change the loss function after each dataset switch. In fact, it does not require any a priori knowledge of the dataset presentation sequence, since these changes in data distribution are automatically inferred. We have demonstrated that VASE can learn a disentangled representation of a sequence of datasets. It does so without experiencing catastrophic forget-

ting and by dynamically allocating spare capacity to represent new information. It resolves ambiguity in a manner that is analogous to the categorical perception characteristic of biological intelligence. Most importantly, VASE allows for semantically meaningful sharing of latents between different datasets, which enables it to perform cross-domain inference and imagination-driven exploration. Taken together, these properties make VASE a promising algorithm for learning representations that are conducive to subsequent robust and data-efficient RL policy learning.

3.4 Related work

The existing approaches to continual learning can be broadly separated into three categories: data-, architecture- or weights-based. The data-based approaches augment the training data on a new task with the data collected from the previous tasks, allowing for simultaneous multi-task learning on IID data (Espeholt et al., 2018; Robins, 1995; Ratcliff, 1990; McClelland et al., 1995; Furlanello et al., 2016). The architecture-based approaches dynamically augment the network with new task-specific modules, which often share intermediate representations to encourage positive transfer (Rusu et al., 2016; Parisotto et al., 2015; Ruvolo and Eaton, 2013). Both of these types of approaches, however, are inefficient in terms of the memory requirements once the number of tasks becomes large. The weights-based approaches do not require data or model augmentation. Instead, they prevent catastrophic forgetting by slowing down learning in the weights that are deemed to be important for the previously learnt tasks (Kirkpatrick et al., 2017a; Zenke et al., 2017; Nguyen et al., 2018). This is a promising direction, however, its application is limited by the fact that it typically uses knowledge of the task presentation schedule to update the loss function after each switch in the data distribution.

Most of the continual learning literature, including all of the approaches discussed above, have been developed in task-based settings, where representations are learnt implicitly. While deep networks learn well in such settings (Achille and Soatto, 2017; Shwartz-Ziv and Tishby, 2017), this often comes at a cost of reduced positive transfer. This is because the implic-

itly learnt representations often overfit to the training task by discarding information that is irrelevant to the current task but may be required for solving future tasks (Achille and Soatto, 2017, 2018; Achille and Soatto, 2018a; Shwartz-Ziv and Tishby, 2017; Higgins et al., 2017c). The acquisition of useful representations of complex high-dimensional data without task-based overfitting is a core goal of unsupervised learning. Past work (Achille and Soatto, 2018; Alemi et al., 2016; Higgins et al., 2017a) has demonstrated the usefulness of information-theoretic methods in such settings. These approaches can broadly be seen as efficient implementations of the Minimum Description Length (MDL) principle for unsupervised learning (Rissanen, 1978; Grünwald, 2007). The representations learnt through such methods have been shown to help in transfer scenarios and with data efficiency for policy learning in the Reinforcement Learning (RL) context (Higgins et al., 2017c). These approaches, however, do not immediately generalise to non-stationary data. Indeed, life-long unsupervised representation learning is relatively under-developed (Shin et al., 2017; Seff et al., 2017; Nguyen et al., 2018). The majority of recent work in this direction has concentrated on implicit generative models (Shin et al., 2017; Seff et al., 2017), or non-parametric approaches (Milan et al., 2016). Since these approaches do not possess an inference mechanism, they are unlikely to be useful for subsequent task or policy learning. Furthermore, none of the existing approaches explicitly investigate meaningful sharing of latent representations between environments.

CHAPTER 4

The Information in the Weights

In the previous chapter, we have discussed what is the minimal amount of information about the current inputs that the network should extract in order to successfully solve a task. In this chapter we study instead what information about past experiences, *i.e.*, the training set, should be extracted and encoded in the parameters of the model. As mentioned in the Introduction, memorizing each training datum is both wasteful and inefficient. Rather, we want to be able to extract and store the "structure" of the task, while leaving out details that may not be relevant for future inference. This raises the natural question: What should be considered important structure, and what is instead a nuisance? The distinction may not always be clear: For example, given a large object dataset, we can learn that, unlike planes, car do not have wings, clearly important information for the task of object classification. We may also discover that the 1917 Ford Model T has a curved hood, unlike the 1915 model. While this difference is not a nuisance (it will be shared by all future examples), it is arguably not an important notion to learn in order to solve a simple object classification task.

In this chapter, we will follow a formalism introduced by Kolmogorov to formally define what we mean by *structure of a task* in term of the trade-off between the complexity of the model and the error committed on the dataset. Unfortunately, Kolmogorov's original approach is based on Kolmogorov complexity, which is not amenable for deep learning. Rather, we introduce a definition of complexity, which we call *Information in the Weights* (IIW), which generalizes several frameworks (Kolmogorov complexity, Rissanen's stochastic complexity, Fisher's Information, Bayesian inference), and can be easily instantiated for DNNs. We then show that a deep network can, in principle, be trained to learn the structure of the task without memorizing by minimizing a variational loss function, which assumes

the form of a *Information Bottleneck Lagrangian for the weights of the network*. While in principle distinct from the activations' IBL introduced in the previous chapter, we show that, in a Deep Network, the former bounds the latter. This shows that in a Deep Network a model that learns to solve a task without memorizing is bound to also learn an invariant representation of the input data, even if this is not actually required or explicitly encouraged: it is an emergent property. Finally, we notice that IIW bounds the generalization error of the network through a PAC-Bayes bound.

4.1 Information in the Weights and Structure Function of a Task

As before, we call task any a random variable we want to infer given an observation x, and given a *training set* $\mathcal{D}$ consisting of data samples x_i and their corresponding task values y_i, $\mathcal{D}_N = \{x_i, y_i\}_{i=1}^N$. A task may therefore be identified with the given dataset $\mathcal{D}$, or with an approximation of the posterior density $p(y|x)$ given the dataset. Of course, without additional hypotheses, there is no unique or right solution to the inference problem: Any label assignment on unseen data may, in principle, be correct.

Moreover, even the label of seen data may be ambiguous: if the dataset is composed by a sequence of uniformly random labels, should we output the memorized labels, or correspondingly the posterior $p(y = y_i|x_i) = 1$, or should we rather output a uniform posterior, *i.e.*, $p(y = y_j|x_i) = 1/|\mathcal{Y}|$ for each $y_j \in \mathcal{Y}$? Notice that in one case, our task has a very complex structure, since we need to memorize several labels, while in the other, the description of the task would be trivial.

An elegant approach to these issues was proposed by Kolmogorov (Vereshchagin and Vitányi, 2004), which we present through a convenient generalization inspired by the bits-back argument of Hinton and Van Camp (1993). Suppose we have a class of models $p_w(y|x)$ parametrized by $w \in \mathcal{W}$ and we need to encode the labels of the dataset. Naively encoding each label independenly is clearly suboptimal in general, as we may instead choose one model $p_{w^*}(y|x)$ that correctly predicts the labels and transmit only the the parameters w^* and the error correction code for the labels that are predicted incorrectly. However, if the

parameters w are continuous, we have an additional problem of how to encode them with finite information. Intuitively, as long as they are encoded with sufficient precision, any mistake due to the loss of precision can be considered part of the incorrect labels. This argument is formalized by Hinton and Van Camp (1993): let $p(w)$ a prior distribution used to encode the weights, and $q(w|\mathcal{D})$ be a "posterior" distribution, that we use to encode the weights to the desired level of precision. For example, $q(w|\mathcal{D}) = \delta_{w^*}$ may be used to encode the weights w^* without any loss of precision, while a gaussian distribution $q(w|\mathcal{D}) \sim N(w^*, \Sigma)$ can represent the weights to a lower precision level given by the prescribed variance Σ. Then, the cost of encoding the dataset is

$$C(\mathcal{D}; q(w|\mathcal{D})) = \underbrace{\mathbb{E}_{w\sim q(w|\mathcal{D})}[L_{\mathcal{D}}(p_w(y|x)]}_{\text{Label reconstruction error}} \quad + \quad \underbrace{\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w))}_{\text{Information in the Weights}} .$$

The first term is the expected cross-entropy loss of models from the distribution $q(w|\mathcal{D})$ on the dataset $\mathcal{D}$. The second term denotes the encoding length of the model $p_w(y|x)$, and we refer to it as *Information in the Weights* (IIW). Depending on the choice of the prior and posterior distribution, this term relates to several measures of information, as we will see in the next section.

Naïvely, we may think that $C(\mathcal{D}) = \min_{q(w|\mathcal{D})} C(\mathcal{D}; q(w|\mathcal{D}))$ is the complexity of the dataset $\mathcal{D}$, and that the corresponding information in the weights $\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w))$ represents the amount of structure contained in the dataset. However, this definition also fails to capture the fact that, for each task, there is a critical performance level range below which many models can perform despite failing to capture the structure in the data. One can easily build models that can classify "airplane" from "fireplace," for instance by counting the number of blue pixels in the image. However, these models are unlikely to be very precise. To reach a small error, say 1%, one has to learn the structure of the dataset: what makes an airplane different from a fireplace even if the latter is painted blue.

Solving a task at different level of precision requires learning different amount of structure, and the complexity of different task can be compared only by looking at the complexity of solving them at each level of precision. For this reason, we introduce the *structure function*

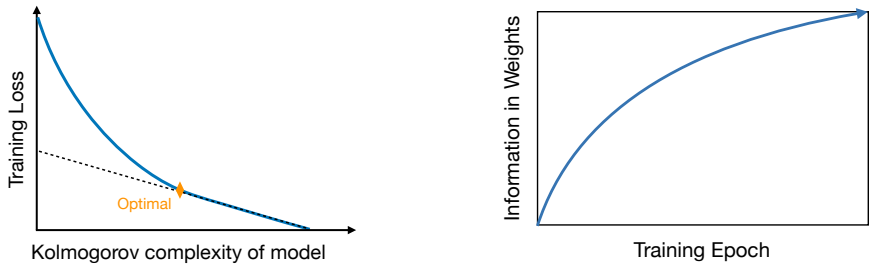


Figure 4.1: Speculative trends for the tradeoff between complexity and fidelity **(left)** and for the information in the weights **(right)**. On the left, at zero bits the model's prediction is at chance level. As the available resources increase, the model can learn simple features (few bits) that improve performance rapidly. As we push for more precision, the model is forced to extract and store more structure from the dataset. Eventually, to bring the loss to zero, the model is forced to memorize all samples that it cannot explain, leading to a straight line with low pendence (memorizing is the worst trade-off between used resources and reduction in error). The point where training loss meets the line marks the transition between underfitting and overfitting. **(Right)** As for the information in the weights, a model starts with no information and, as we proceed through training epochs, we expect the information in the weights to increase until it saturates, while the training loss decreases and stabilizes. While the figure on the left closely resembles what we observe in reality, the figure on the right, is not quite accurate (Chapter 6).

of a dataset (task) $\mathcal{D}$ as:

$$S_{\mathcal{D}}(t) = \min_{\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w)) < t} \mathbb{E}_{w \sim q(w|\mathcal{D})}[L_{\mathcal{D}}(p_w(y|x))], \tag{4.1}$$

This definition has the important quality of recognizing that there is no single definition of complexity of a task: depending on the level of accuracy we want to achieve, different complexities are required.

We will discuss at length how to actually compute the complexity of the task and the information in the weights. But before doing so, let us speculate how these functions may look like. The cartoon in Figure 4.1 (left) shows how we might expect the training loss to decrease as we increase the coding length, and on the right how the information in the weights may change during training. As we will see, reality follows closely for the former, but for the latter is not quite accurate, as we will see extensively in Chapter 6.

4.2 Relation to Shannon, Fisher, and Kolmogorov

The choice of encoding, or prior, $p(w)$ in the definition of task complexity is arbitrary. We consider three special (and extreme) cases: In one, we use an improper, uninformative prior, oblivious of the dataset, and show that the Information in the Weights reduces to the log-determinant of Fisher's Information, up to a constant. In the second case, we use an "adapted" prior, that removes the dependency on the particular dataset by averaging over all possible dataset, that is, $q(w) = \mathbb{E}_{\mathcal{D}}[q(w|\mathcal{D})]$. In this case, we show that the Information in the Weights reduces to Shannon's Mutual Information $I(w;\mathcal{D})$. In the third case, when choosing the "universal prior" of computable functions, we obtain Kolmogorov's Complexity.

Proposition 4.2.1 (Shannon Information in the Weights)**.** *Assume the dataset $\mathcal{D}$ is sampled from a distribution $p(\mathcal{D})$ and let the outcome of training on a sampled dataset $\mathcal{D}$ be described by a distribution $q(w|\mathcal{D})$. Then prior $p(w)$ minimizing the coding length is $p(w) = q(w) = \mathbb{E}_{\mathcal{D}}[q(w|\mathcal{D})]$, and the (expected) Information in the Weights for a task $\mathcal{D}$ is given by*

$$\mathbb{E}_{\mathcal{D}}[\mathrm{KL}(q(w|\mathcal{D}) \parallel q(w))] = I(w;\mathcal{D}) \tag{4.2}$$

where $I(w;\mathcal{D})$ is Shannon's Mutual Information between the weights and the dataset.

Note that, in this case, the prior $p(w) = q(w)$ is optimal given the choice of the training algorithm (e.g., the map $A : \mathcal{D} \to q(w|\mathcal{D})$), and the particular class of distribution of training tasks $p(\mathcal{D})$, which, however, we do not know in general, as we are often given a single dataset to train.

Proof. For a fixed training algorithm $A : \mathcal{D} \mapsto q(w|\mathcal{D})$, we want to find $p(w)$ that minimizes the expected complexity of the data:

$$\begin{aligned} p^*(w) &= \arg\min_{p(w)} \mathbb{E}_{\mathcal{D}}[C(\mathcal{D})] \\ &= \arg\min_{p(w)} \mathbb{E}_{\mathcal{D}}[L_{\mathcal{D}}(p_w(y|x))] + \mathbb{E}_{\mathcal{D}}[\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w))] \end{aligned}$$

Notice that only the second term depends on $p(w)$. Let $q(w) = \mathbb{E}_{\mathcal{D}}[p(w|\mathcal{D})]$ be the marginal distribution of w, averaged over all possible training datasets. Notice that we have the

identity

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}[\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w))] &= \mathbb{E}_{\mathcal{D}}[\mathrm{KL}(q(w|\mathcal{D}) \parallel q(w))] + \mathbb{E}_{\mathcal{D}}[\mathrm{KL}(q(w) \parallel p(w))] \\ &= I(w;\mathcal{D}) + \mathrm{KL}(q(w) \parallel p(w)).\end{aligned}$$

Hence, using the fact that the KL divergence is always positive, the optimal "adapted" prior that minimizes the expected encoding length of $\mathcal{D}$ is given by $p^*(w) = q(w)$, that is, the marginal distribution of w over all trainings and datasets. With this choice, we obtain finally $\mathbb{E}_{\mathcal{D}}[\mathrm{KL}(q(w|\mathcal{D}) \parallel p^*(w))] = I(w;\mathcal{D})$. □

Computing the marginal distribution $q(w)$ over all possible datasets is not realistic. However, it is comforting to know that the unrealistic special case of the Information in the Weights we defined relates to Shannon's Information. A more practical choice is to pick both $q(w|\mathcal{D}) \sim N(w_0, \Sigma)$ and $p(w) = N(0, \sigma_0^2 I)$ to be Gaussian, so the Information in the Weights has a closed-form solution, which relates to the Fisher Information Matrix.

Proposition 4.2.2 (Fisher Information)**.** *Let w_0 be a local minimum of the cross-entropy loss $L_{\mathcal{D}}(w)$, and assume a gaussian posterior $q(w|\mathcal{D}) \sim N(w_0, \Sigma)$ and prior $p(w) = N(0, \sigma_0^2 I)$. Then, the optimal choice of Σ is*

$$\Sigma^* = \left(F(w_0) + \frac{1}{2\sigma_0^2} I\right)^{-1},$$

where $F(w_0)$ is the Fisher Information Matrix computed at w_0. In particular, when $\sigma_0 \to \infty$, i.e., when the prior becomes uniform, we have

$$\Sigma^* \xrightarrow{\sigma_0 \to \infty} F(w_0)^{-1}, \text{ and } \mathrm{KL}(q(w|\mathcal{D}) \parallel p(w)) \xrightarrow{\sigma_0 \to \infty} \frac{1}{2} \log |F(w_0)| + c(\sigma_0),$$

where $c(\sigma_0)$ is an additive constant that depends on σ_0. That is, the information in the weights is given by the log-determinant of the Fisher Information Matrix. Notice that $\log |F|$ has a natural interpretation as a measure of the curvature of the loss function at that point.

Proof. Since both prior and posterior are gaussian, using the known close form expression for the KL divergence of two gaussians we have

$$\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w)) = \frac{1}{2}\left[\sigma_0^{-2} w_0^T w_0 + \mathrm{tr}(\sigma_0^{-2}\Sigma) - \log |\sigma_0^{-2}\Sigma| - k\right].$$

To find the optimal variance Σ of the posterior, we need to solve:

$$\begin{aligned}\Sigma^* &= \arg\min_{\Sigma} C(\mathcal{D}) \\ &= \arg\min_{\Sigma} \mathbb{E}_{w\sim q(w|\mathcal{D})}[L_{\mathcal{D}}(w)] + \mathrm{KL}(q(w|\mathcal{D}) \parallel p(w))\end{aligned}$$

Expanding to the second order $L_{\mathcal{D}}(w)$, we have:

$$\begin{aligned}\mathbb{E}_{w\sim q(w|\mathcal{D})}[L_{\mathcal{D}}(w)] &= \mathbb{E}_{w\sim q(w|\mathcal{D})}[L_{\mathcal{D}}(w_0) + \nabla L_{\mathcal{D}}(w - w_0) + \frac{1}{2}(w - w_0)^T H(w - w_0)] \\ &= L_{\mathcal{D}}(p_{w_0}(y|x) + \mathrm{tr}(\Sigma H),\end{aligned}$$

where H is the Hessian of $L_{\mathcal{D}}$ computed in $w = w_0$. Putting this approximation back in the expression for $C(\mathcal{D})$, and optimizing for Σ using the known formula for the KL divergence, we obtain

$$\Sigma^* = \big(H + \frac{1}{2\sigma_0^2} I\big)^{-1}.$$

Notice that, when using the cross-entropy loss, the Hessian $H(w_0)$ of the loss function at a stationary point of the training procedure coincides with the Fisher Information Matrix $F(w_0)$ (Martens and Grosse, 2015).Hence, we have that

$$\Sigma^* \xrightarrow{\sigma_0 \to \infty} F^{-1},$$

that is, the optimal variance tends to the inverse of the Fisher Information Matrix when the prior becomes uniform. Notice that this is indeed in accordance with Cramér-Rao bound. Hence, for the optimal choice of the variance, we have

$$\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w)) \xrightarrow{\sigma_0 \to \infty} \frac{1}{2} \log |F| + c(\sigma_0),$$

where $c(\sigma_0)$ is an additive constant that goes to infinity as $\sigma_0 \to \infty$. □

Remark 4.2.3 (Flat minima have low information)**.** Additional indirect empirical evidence is provided by the fact that some variants of SGD (Chaudhari et al., 2017) bias the optimization toward "flat minima", that are local minima whose Hessian and hence their Fisher (Martens (2014)), has mostly small eigenvalues. Under the previous model, these minima corresponds to minima with low information, as suggested early on by Hochreiter and

Schmidhuber (1997): As we will see later, these implies that flat minima have better generalization properties (Section 4.4) and that the associated representation of the data is more insensitive to nuisances and more disentangled (Section 4.5).

Finally, we consider the algorithmic setting. Recall that, for a given string w, we have $p(w) := \sum_p 2^{-|p|} \simeq 2^{-K(w)}$, where the sum is over all programs p that outputs w, $|p|$ is the length of the program, and $K(w)$ is the Kolmogorov complexity of w, that is, the length of the shortest program that output w.

Proposition 4.2.4 (Kolmogorov Complexity of the Weights)**.** *Let $p(w)$ be the universal prior; then the Information In the Weights equals the Kolmogorov Complexity of the weights $K(w)$.*

Proof. Using $p(w)$ as the prior, when the posterior is a Dirac delta $q(w|\mathcal{D}) = \delta_{w^*}$ we trivially have $\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w)) = -\log(2^{-K(w)}) = K(w)$. □

4.3 Information Bottleneck Lagrangian for the Weights

In Section 4.1 we have introduced the structure function of a dataset $\mathcal{D}$ as

$$S_{\mathcal{D}}(t) = \min_{\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w)) < t} \mathbb{E}_{w \sim q(w|\mathcal{D})}[L_{\mathcal{D}}(p_w(y|x))].$$

This function allows to study how much structure is present in the dataset at various level of precision. More importantly for applications, it allows us to find solutions that fit the data without overfitting.

In order to compute the structure function, we may consider the lagrangian associated to the minimization problem, which is given by:

$$C_\beta(\mathcal{D}; P, Q) = \mathbb{E}_{w \sim q(w|\mathcal{D})}[L_{\mathcal{D}}(p_w(y|x))] + \beta \,\mathrm{KL}(q(w|\mathcal{D}) \parallel p(w)) \tag{4.3}$$

Notice that, when picking the adapted prior, this reduces to:

$$\mathcal{L}(q(w|\mathcal{D})) = \mathbb{E}_{w \sim q(w|\mathcal{D})}[L_{\mathcal{D}}(p_w(y|x))] + \beta I(w; \mathcal{D}), \tag{4.4}$$

which is precisely an Information Bottleneck Lagrangian, as the we encountered in Section 2.3, however, that was an Information Bottleneck Lagrangian for the activations of the network, which limits the mutual information between input and representation and is related to nuisance invariance. On the other hand, this IBL concerns the weights of the network, seen as a representation of the training data, and relates to the fitting the data without overfitting.

In principle the two IBL do not need to be related in any way, aside from the similar form. However, in the next section, we prove one of our main results, that in the case of deep networks one the IBL of the weights control the one for activations. Therefore, invariance and disentanglement emerge naturally when training a network with implicit (SGD) or explicit (IB Lagrangian) regularization to avoid overfitting, and are related to flat minima.

Before, proving these results, we should notice that, when $\beta = 1$, eq. (4.3) reduces to the ELBO loss used in variational inference. In fact, assuming an asymptotically optimal model, $\beta = 1$ is the value of the IBL that recovers all the structure in the data without memorizing. It may be tempting then to always use $\beta = 1$. However, there are two factors to consider: (a) the task may be very complex to solve at $\beta = 1$, while much simpler solution for higher β may still lead to good performance, and (b) the model class may not be asymptotically optimal, and may require a different value of β to reach the critical point.

The connection with the classical ELBO however provides several efficient ways to minimize the IBL for the weights. In particular, it can be easily minimized in standard deep network using Stochastic Gradient Variational Bayes (SGVB) using a slight variation of the framework of Kingma et al. (2015).

4.4 Generalization guarantees through the PAC-Bayes bound

The Lagrangian $C_\beta(\mathcal{D}; P, Q)$ admits another interpretation as an upper-bound to the test error, as shown by the PAC-Bayes test error bound:

Theorem 4.4.1 (McAllester (2013))**.** *Assume the dataset* $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ *is sampled i.i.d.*

from a distribution $p(y, x)$, and assume that the per-sample loss used to train is bounded by $L_{max} = 1$ (we can reduce to this case by clipping and rescaling the loss). For any fixed $\beta > 1/2$, prior $P(w)$, and weight distribution $Q(w|\mathcal{D})$, with probability at least $1-\delta$ over the sample of $\mathcal{D}$, we have:

$$L_{test}(Q) \leq \frac{1}{N(1-\frac{1}{2\beta})}\Big[\mathbb{E}_{w\sim Q(w|\mathcal{D})}[L_{\mathcal{D}}(p_w)] + \beta\,\mathrm{KL}(Q \parallel P) + \beta \log\frac{1}{\delta}\Big] \tag{4.5}$$

$$= \frac{1}{N(1-\frac{1}{2\beta})}\Big[C_\beta(\mathcal{D}; P, Q) + \beta\log\frac{1}{\delta}\Big]. \tag{4.6}$$

where $L_{test}(Q) := \mathbb{E}_{x,y\sim p(x,y)}[\mathbb{E}_{w\sim Q}[p_w(y|x)]]$ is the expected per-sample test error that the model incurs using the weight distribution $Q(w|\mathcal{D})$.

Proof. First, we consider the case in which $\dim(z) = 1$, and so $w := W$ is a single row vector. By hypothesis, $q(z)$ is approximately Gaussian, with mean and variance

$$\begin{aligned}
\mu_1 &:= \mathbb{E}[z] = \mathbb{E}[\sum_i \epsilon_i \hat{w}_i x_i] = \sum_i \hat{w}_i \mathbb{E}[x_i] = \hat{w}\cdot\mathbb{E}[x] \\
\sigma_1^2 &:= \mathrm{var}[z] = \mathbb{E}[(\sum_i \epsilon_i \hat{w}_i x_i)^2] - (\mathbb{E}[\sum_i \epsilon_i \hat{w}_i x_i])^2, \\
&= \mathbb{E}[\sum_{i,j} \epsilon_i\epsilon_j \hat{w}_i\hat{w}_j x_i x_j] - \sum_{i,j}\hat{w}_i\hat{w}_j\mathbb{E}[x_i]\mathbb{E}[x_j] \\
&= \tilde{\alpha}\sum_i \hat{w}_i^2\mathbb{E}[x_i]^2 + \sum_{i,j}\hat{w}_i\hat{w}_j\left(\mathbb{E}[x_i x_j] - \mathbb{E}[x_i]\mathbb{E}[x_j]\right) \\
&= \tilde{\alpha}\hat{w}^2\cdot\mathbb{E}[x^2] + \hat{w}\cdot\mathrm{Cov}(x)\hat{w}.
\end{aligned}$$

A similar computation gives us mean and variance of $q(z|x)$:

$$\begin{aligned}
\mu_0 &:= \mathbb{E}[z|x] = \hat{w}\cdot x, \\
\sigma_0^2 &:= \mathrm{var}[z|x] = \tilde{\alpha}\hat{w}^2\cdot x^2.
\end{aligned}$$

Since we are assuming $\dim(z) = 1$, we trivially have $\mathrm{TC}(z) = 0$, so we are only left with

$I(z;x)$ which is given by

$$\begin{aligned} I(z;x) &= \mathbb{E}_x \,\mathrm{KL}(q(z|x) \parallel q(z)) \\ &= \mathbb{E}_x \,\mathrm{KL}(\mathcal{N}(\mu_0, \sigma_0^2) \parallel \mathcal{N}(\mu_1, \sigma_1^2)) \\ &= \frac{1}{2}\mathbb{E}_x \frac{\tilde{\alpha}\hat{w}^2 \cdot x^2 + (\hat{w} \cdot x - \hat{w} \cdot \mathbb{E}[x])^2}{\sigma_1^2} - 1 - \log \frac{\sigma_0^2}{\sigma_1^2} \\ &= -\frac{1}{2}\mathbb{E}_x \log \frac{\tilde{\alpha}\hat{w}^2 \cdot x^2}{\hat{w} \cdot \mathrm{Cov}(x)\hat{w} + \tilde{\alpha}\hat{w}^2 \cdot \mathbb{E}[x^2]}. \end{aligned}$$

Now, for the general case of $\dim(z) \geq 1$, notice that

$$\begin{aligned} I(\mathbf{z};\mathbf{x}) + \mathrm{TC}(\mathbf{z}) &= \mathbb{E}_x \,\mathrm{KL}(\prod_k q(z_i|\mathbf{x}) \parallel \prod_k q(z_i)) \\ &= \sum_{i=1}^{\dim(z)} \mathbb{E}_x \,\mathrm{KL}(q(z_i|\mathbf{x}) \parallel q(z_i)), \end{aligned}$$

where $q(z_i)$ is the marginal of the k-th component of z. We can then use the previous result for each component separately, and sum everything to get the desired identity. □

Hence, we see that minimizing the Lagrangian $C_\beta(\mathcal{D}; P, Q)$ can be interpreted as minimizing an upper-bound on the test error of the model, rather than directly minimizing the train error. This is in accordance with the intuition developed earlier, that minimizing $C_\beta(\mathcal{D}; P, Q)$ forces the model to capture the structure of the data. It is also interesting to consider the following bound on the expectation over the sampling of $\mathcal{D}$ (McAllester, 2013, Theorem 4):

$$\mathbb{E}_{\mathcal{D}}[L_{\text{test}}(Q(w|\mathcal{D}))] \leq \frac{1}{N(1 - \frac{1}{2\beta})}\Big[\mathbb{E}_{\mathcal{D}}[L_{\mathcal{D}}(Q(w|\mathcal{D}))] + \beta\, \mathbb{E}_{\mathcal{D}}[\mathrm{KL}(Q(w|\mathcal{D}) \parallel P)]\Big].$$

As we have seen in Proposition 4.2.1, for the optimal choice of prior P minimizing the bound, we have $\mathbb{E}_{\mathcal{D}}[\mathrm{KL}(Q \parallel P)] = I(w;\mathcal{D})$. Hence, the Shannon Information that the weights of the model have about the dataset $\mathcal{D}$ is the measure of complexity that gives (on expectation) the strongest generalization bound. This has also been noted in Achille and Soatto (2017). In Dziugaite and Roy (2017), a non-vacuous generalization bound is computed for DNNs, using (non-centered and non-isotropic) Gaussian prior and posterior distributions.

4.5 Emergence of invariance and disentanglement during training

The following proposition gives the fundamental link in our model between information in the weights, and hence flatness of the local minima, minimality of the representation, and disentanglement.

Proposition 4.5.1. *Let $z = Wx$, and assume as before $W = \epsilon \odot \hat{W}$, with $\epsilon_{i,j} \sim \log \mathcal{N}(-\alpha_i/2, \alpha_i)$. Further assume that the marginals of $p(z)$ and $p(z|x)$ are both approximately Gaussian (which is reasonable for large $\dim(x)$ by the Central Limit Theorem). Then,*

$$I(z;x) + \mathrm{TC}(z) = -\frac{1}{2} \sum_{i=1}^{\dim(z)} \mathbb{E}_x \log \frac{\tilde{\alpha}_i \hat{W}_i^2 \cdot x^2}{\hat{W}_i \cdot \mathrm{Cov}(x) \hat{W}_i + \tilde{\alpha}_i \hat{W}_i^2 \cdot \mathbb{E}(x^2)}, \tag{4.7}$$

where W_i denotes the i-th row of the matrix W, and $\tilde{\alpha}_i$ is the noise variance $\tilde{\alpha}_i = \exp(\alpha_i) - 1$. In particular, $I(z;x) + \mathrm{TC}(z)$ is a monotone decreasing function of the weight variances α_i.

Proof. First, we consider the case in which $\dim(z) = 1$, and so $w := W$ is a single row vector. By hypothesis, $q(z)$ is approximately Gaussian, with mean and variance

$$\begin{aligned}
\mu_1 &:= \mathbb{E}[z] = \mathbb{E}[\sum_i \epsilon_i \hat{w}_i x_i] = \sum_i \hat{w}_i \mathbb{E}[x_i] = \hat{w} \cdot \mathbb{E}[x] \\
\sigma_1^2 &:= \mathrm{var}[z] = \mathbb{E}[(\sum_i \epsilon_i \hat{w}_i x_i)^2] - (\mathbb{E}[\sum_i \epsilon_i \hat{w}_i x_i])^2, \\
&= \mathbb{E}[\sum_{i,j} \epsilon_i \epsilon_j \hat{w}_i \hat{w}_j x_i x_j] - \sum_{i,j} \hat{w}_i \hat{w}_j \mathbb{E}[x_i] \mathbb{E}[x_j] \\
&= \tilde{\alpha} \sum_i \hat{w}_i^2 \mathbb{E}[x_i]^2 + \sum_{i,j} \hat{w}_i \hat{w}_j \left(\mathbb{E}[x_i x_j] - \mathbb{E}[x_i] \mathbb{E}[x_j] \right) \\
&= \tilde{\alpha} \hat{w}^2 \cdot \mathbb{E}[x^2] + \hat{w} \cdot \mathrm{Cov}(x) \hat{w}.
\end{aligned}$$

A similar computation gives us mean and variance of $q(z|x)$:

$$\begin{aligned}
\mu_0 &:= \mathbb{E}[z|x] = \hat{w} \cdot x, \\
\sigma_0^2 &:= \mathrm{var}[z|x] = \tilde{\alpha} \hat{w}^2 \cdot x^2.
\end{aligned}$$

Since we are assuming $\dim(z) = 1$, we trivially have $\mathrm{TC}(z) = 0$, so we are only left with

$I(z;x)$ which is given by:

$$\begin{aligned} I(z;x) &= \mathbb{E}_x \operatorname{KL}(q(z|x) \parallel q(z)) \\ &= \mathbb{E}_x \operatorname{KL}(\mathcal{N}(\mu_0, \sigma_0^2) \parallel \mathcal{N}(\mu_1, \sigma_1^2)) \\ &= \frac{1}{2}\mathbb{E}_x \frac{\tilde{\alpha}\hat{w}^2 \cdot x^2 + (\hat{w} \cdot x - \hat{w} \cdot \mathbb{E}[x])^2}{\sigma_1^2} - 1 - \log \frac{\sigma_0^2}{\sigma_1^2} \\ &= -\frac{1}{2}\mathbb{E}_x \log \frac{\tilde{\alpha}\hat{w}^2 \cdot x^2}{\hat{w} \cdot \operatorname{Cov}(x)\hat{w} + \tilde{\alpha}\hat{w}^2 \cdot \mathbb{E}[x^2]}. \end{aligned}$$

Now, for the general case of $\dim(z) \geq 1$, notice that

$$\begin{aligned} I(\mathbf{z};\mathbf{x}) + \operatorname{TC}(\mathbf{z}) &= \mathbb{E}_x \operatorname{KL}(\prod_k q(z_i|\mathbf{x}) \parallel \prod_k q(z_i)) \\ &= \sum_{i=1}^{\dim(z)} \mathbb{E}_x \operatorname{KL}(q(z_i|\mathbf{x}) \parallel q(z_i)), \end{aligned}$$

where $q(z_i)$ is the marginal of the k-th component of z. We can then use the previous result for each component separately, and sum everything to get the desired identity. □

The above identity is difficult to apply in practice, but with some additional hypotheses, we can derive a clearer uniform tight bound on $I(z;x) + \operatorname{TC}(z)$.

Proposition 4.5.2 (Uniform bound for one layer)**.** *Let $z = Wx$, where $W = \epsilon \odot \hat{W}$, where $\epsilon_{i,j} \sim \log\mathcal{N}(-\alpha/2, \alpha)$; assume that the components of x are uncorrelated, and that their kurtosis is uniformly bounded.*[1] *Then, there is a strictly increasing function $g(\alpha)$ s.t. we have the uniform bound*

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

where $c = O(1/\dim(x)) \leq 1$, $g(\alpha) = -\log(1 - e^{-\alpha})/2$ and α is related to $\tilde{I}(w;\mathcal{D})$ by $\alpha = \exp\{-I(W;\mathcal{D})/\dim(W)\}$. In particular, $I(x;z) + TC(z)$ is tightly bounded by $\tilde{I}(W;\mathcal{D})$ and increases strictly with it.

[1] *This is a technical hypothesis, always satisfied if the components x_i are IID, (sub-)Gaussian, or with uniformly bounded support.*

Proof. To simplify the notation we do the case $\dim z = 1$, the general case being identical. Let $w := W$ be the only row of W. First notice that, since x is uncorrelated, we have

$$\hat{w} \cdot \mathrm{Cov}(x)\hat{w} = \sum_i w_i^2(\mathbb{E}[x_i^2] - \mathbb{E}[x_i]^2) \leq w^2 \cdot \mathbb{E}[x^2]$$

Therefore,

$$\begin{aligned} I(x;z) =& -\frac{1}{2}\mathbb{E}_x \log \frac{\tilde{\alpha}\hat{w}^2 \cdot x^2}{\hat{w} \cdot \mathrm{Cov}(x)\hat{w} + \tilde{\alpha}\hat{w}^2 \cdot \mathbb{E}[x^2]} \\ \leq& -\frac{1}{2}\mathbb{E}_x \log \frac{\tilde{\alpha}\hat{w}^2 \cdot x^2}{(1+\tilde{\alpha})\hat{w}^2 \cdot \mathbb{E}[x^2]} \\ =& \frac{1}{2}\log(1+\tilde{\alpha}^{-1}) - \frac{1}{2}\mathbb{E}_x \log\left[1 + \frac{\hat{w}^2 \cdot (x^2 - \mathbb{E}[x^2])}{\hat{w}^2 \cdot \mathbb{E}[x^2]}\right]. \end{aligned}$$

To conclude, we want to approximate the expectation of the logarithm using a Taylor expansion, but we first need to check that the variance of the term inside the logarithm is low, which is where we need the bound on the kurtosis. In fact, since the kurtosis is bounded, there is some constant C such that for all i

$$\frac{\mathbb{E}(x_i^2 - \mathbb{E}[x_i^2])^2}{\mathbb{E}[x_i^2]^2} \leq C.$$

Now,

$$\begin{aligned} \mathrm{var}\,\frac{\hat{w}^2 \cdot (x^2 - \mathbb{E}[x^2])}{\hat{w}^2 \cdot \mathbb{E}[x^2]} &= \frac{\sum_i \hat{w}_i^4 \mathbb{E}(x^2 - \mathbb{E}[x^2])^2}{\sum_{i,j} \hat{w}_i^2 \hat{w}_j^2 \mathbb{E}[x_i^2]\mathbb{E}[x_j^2]} \\ &\leq C \frac{\sum_i \hat{w}_i^4 \mathbb{E}[x_i^2]^2}{\sum_{i,j} \hat{w}_i^2 \hat{w}_j^2 \mathbb{E}[x_i^2]\mathbb{E}[x_j^2]} \\ &= O(1/\dim(x)). \end{aligned}$$

Therefore, we can conclude

$$I(x;z) \leq \frac{1}{2}\log(1+\tilde{\alpha}^{-1}) + O(1/\dim(x)). \qquad \square$$

The above theorems tells us that whenever we decrease the information in the weights, either by explicit regularization, or by implicit regularization (*e.g.*, using SGD), we automatically improve the minimality, and hence, by Proposition 2.4.1, the invariance, and the disentanglement of the learner representation. In particular, we obtain as a corollary that SGD is biased toward learning invariant and disentangled representations of the data. Using the Markov property of the layers, we can easily extend this bound to multiple layers:

Corollary 4.5.3 (Multi-layer case)**.** *Let W^k for $k = 1, ..., L$ be weight matrices, with $W^k = \epsilon^k \odot \hat{W}^k$ and $\epsilon^k_{i,j} = \log \mathcal{N}(-\alpha^k/2, \alpha^k)$, and let $z_{i+1} = \phi(W^k z_k)$, where $z_0 = x$ and ϕ is any nonlinearity. Then,*

$$I(z_L; x) \leq \min_{k<L} \left\{ \dim(z_k) \left[g(\alpha^k) + 1 \right] \right\}$$

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

Proof. Since we have the Markov chain $x \to z_1 \to \ldots \to z_L$, by the Data Processing Inequality we have $I(z_L; x) \leq \min\{I(z_L; z_{L-1}), I(z_{L-1}; x)\}$. Iterating this inequality, we have

$$I(z_L; x) \leq \min_{k<L} I(z_{k+1}, z_k).$$

Now, notice that $I(z_{k+1}; z_k) \leq I(\phi(W^k z_k); z_k) \leq I(W^k z_k; z_k)$, since applying a deterministic function can only decrease the information. But $I(W^k z_k; z_k)$ is exactly the quantity we bounded in Corollary 4.5.2, leading us to the desired inequality. □

Remark 4.5.4 (Tightness)**.** While the bound in Proposition 4.5.2 is tight, the bound in the multilayer case needs not be. This is to be expected: Reducing the information in the weights creates a bottleneck, but we do not know how much information about x will actually go through this bottleneck. Often, the final layers will let most of the information through, while initial layers will drop the most.

Remark 4.5.5 (Training-test transfer)**.** We note that we did not make any (explicit) assumption about the test set having the same distribution of the training set. Instead, we make the less restrictive assumption of sufficiency: If the test distribution is entirely different from the training one – one may not be able to achieve sufficiency. This prompts interesting questions about measuring the distance between *tasks* (as opposed to just distance between distributions), which we study in Chapter 5.

4.6 Empirical validation

Phase transition from overfitting to underfitting

As pointed out by Zhang et al. (2017), when a standard convolutional neural network (CNN) is trained on CIFAR-10 to fit random labels, the network is able to (over)fit them perfectly. This is easily explained in our framework: It means that the network is complex enough to memorize all the labels but, as we show here, it has to pay a steep price in terms of information complexity of the weights (Figure 4.3) in order to do so. On the other hand, when the information in the weights is bounded using and information regularizer, overfitting is prevented in a theoretically predictable way.

In particular, in the case of completely random labels, we have $I(\mathbf{y}; w|\mathbf{x}, \theta) = I(\mathbf{y}; w) \leq I(w; \mathcal{D})$, where the first equality holds since $\mathbf{y}$ is by construction random, and therefore independent of $\mathbf{x}$ and θ. In this case, the IBL of the weights in eq. (4.4) is an optimal regularizer, and, regardless of the dataset size N, for $\beta > 1$ it should completely prevent memorization, while for $\beta < 1$ overfitting is possible. To see this, notice that since the labels are random, to decrease the classification error by $\log |\mathcal{Y}|$, where $|\mathcal{Y}|$ is the number of possible classes, we need to memorize a new label. But to do so, we need to store more information in the weights of the network, therefore increasing the second term $I(w; \mathcal{D})$ by a corresponding quantity. This trade-off is always favorable when $\beta < 1$, but it is not when $\beta > 1$. Therefore, the theoretically the optimal solution to eq. (4.4) is to memorize all the labels in the first case, and not memorize anything in the latter.

As discussed, for real neural networks we cannot directly minimize eq. (4.4), and we assume a simple Gaussian prior and posterior (see Proposition 4.2.2), which in general provides an upper-bound to the optimal Shannon Information (Proposition 4.2.1). Even so, the empirical behavior of the network trained to minimize the gaussian approximation, shown in Figure 4.2, closely follows this prediction, and for various sizes of the dataset clearly shows a phase transition between overfitting and underfitting near the critical value $\beta = 1$. Notice instead that for real labels the situation is different: The model is still able to overfit when $\beta < 1$, but importantly there is a large interval of $\beta > 1$ where the model can fit the data

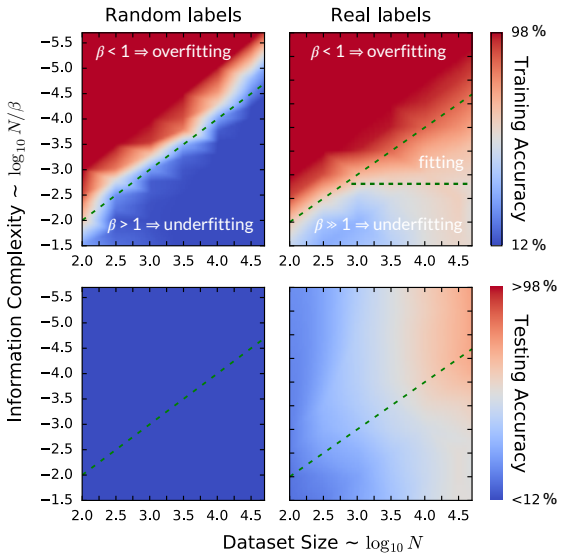


Figure 4.2: **(Left)** The AlexNet model of Zhang et al. (2017) achieves high accuracy (red) even when trained with random labels on CIFAR-10. Using the IB Lagrangian to limit information in the weights leads to a sharp transition to under-fitting (blue) predicted by the theory (dashed line). To overfit, the network needs to memorize the dataset, and the information needed grows linearly. **(Right)** For real labels, the information sufficient to fit the data without overfitting saturates to a value that depends on the dataset, but somewhat independent of the number of samples. Test accuracy shows a uniform blue plot for random labels, while for real labels it increases with the number of training samples, and is higher near the critical regularizer value $\beta = 1$.

without overfitting to it. Indeed, as soon as $\beta N \propto I(w;\mathcal{D})$ is larger than the constant $H(\theta)$, the model trained on real data fits real labels without excessive overfitting (Figure 4.2).

Notice that, based on this reasoning, we expect the presence of a phase transition between an overfitting and an under-fitting regime at the critical value $\beta = 1$ to be largely independent on the network architecture: To verify this, we train different architectures on a subset of 10000 samples from CIFAR-10 with random labels. As we can see on the left plot of Figure 4.3, even very different architectures show a phase transition at a similar value of β. We also notice that in the experiment ResNets has a sharp transition close to the critical β.

In the right plot of Figure 4.3 we measure the quantity information in the weights for different levels of corruption of the labels. To do this, we fix $\beta < 1$ so that the network is able to overfit, and for various level of corruption we train until convergence, and then compute $I(w;\mathcal{D})$ for the trained model. As expected, increasing the randomness of the labels increases the quantity of information we need to fit the dataset. For completely random labels, $I(w;\mathcal{D})$ increases by ~ 3 nats/sample, which the same order of magnitude as the quantity required to memorize a 10-class labels (2.30 nats/sample), as shown in Figure 4.3.

Bias-variance trade-off

The Bias-Variance trade-off is sometimes informally stated as saying that low-complexity models tend to under-fit the data, while excessively complex models may instead overfit, so that one should select an adequate intermediate complexity. This is apparently at odds with the common practice in Deep Learning, where increasing the depth or the number of weights of the network, and hence increasing the "complexity" of the model measured by the number of parameters, does not seem to induce overfitting. Consequently, a number of alternative measures of complexity have been proposed that capture the intuitive bias-variance trade-off curve, such as different norms of the weights (Neyshabur et al., 2015).

From the discussion above, we have seen that the quantity of information in the weights, or alternatively its computable gaussian upper-bound $\mathrm{KL}(N(w,\Sigma) \parallel N(0,\lambda^2 I))$ (Proposition 4.2.2), also provides a natural choice to measure model complexity in relation to

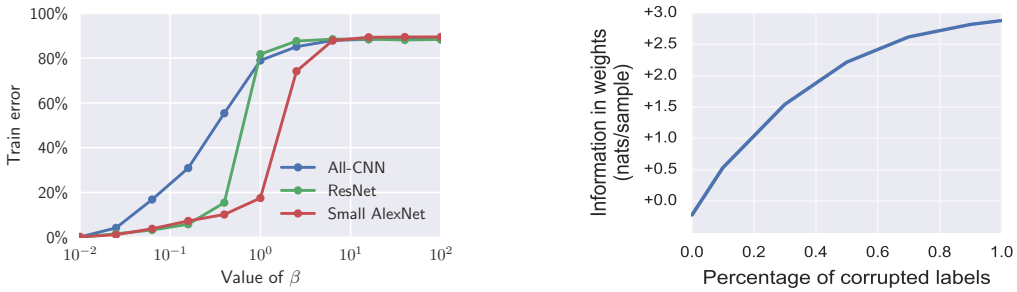


Figure 4.3: **(Left)** Plot of the training error on CIFAR-10 with random labels as a function of the parameter β for different models (see the appendix for details). As expected, all models show a sharp phase transition from complete overfitting to under-fitting before the critical value $\beta = 1$. **(Right)** We measure the quantity of information in the weights necessary to overfit as we vary the percentage of corrupted labels under the same settings of Figure 4.2. To fit increasingly random labels, the network needs to memorize more information in the weights; the increase needed to fit entirely random labels is about the same magnitude as the size of a label (2.30 nats/sample).

overfitting. In particular, we have already seen that models need to store increasingly more information to fit increasingly random labels (Figure 4.3). In Figure 4.4 we show that by controlling $I(w; \mathcal{D})$, which can be done easily by modulating β, we recover the right trend for the bias-variance tradeoff, whereas models with too little information tend to under-fit, while models memorizing too much information tend to overfit.

Nuisance invariance

Corollary 4.5.3 shows that by decreasing the information in the weights $I(w; \mathcal{D})$, which can be done for example using eq. (4.4), the learned representation will be increasingly minimal, and therefore insensitive to nuisance factors n, as measured by $I(z; n)$. Here, we adapt a technique from the GAN literature (Sønderby et al., 2017) that allows us to explicitly measure $I(z; n)$ and validate this effect, provided we can sample from the nuisance distribution $p(n)$ and from $p(x|n)$; that is, if given a nuisance n we can generate data x affected by that

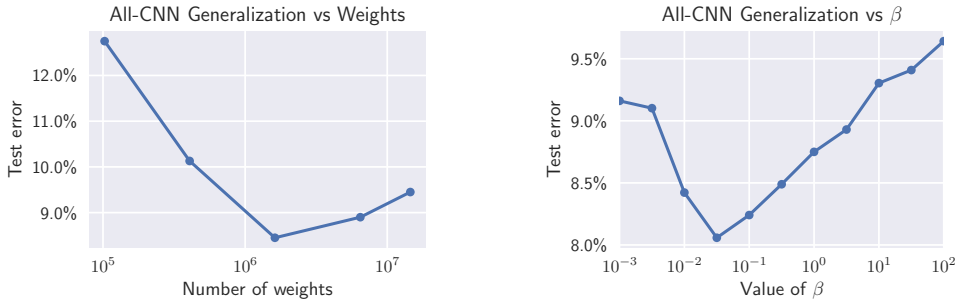


Figure 4.4: Plots of the test error obtained training the All-CNN architecture on CIFAR-10 (no data augmentation). **(Left)** Test error as we increase the number of weights in the network using weight decay but without any additional explicit regularization. Notice that increasing the number of weights the generalization error plateaus rather than increasing. **(Right)** Changing the value of β, which controls the amount of information in the weights, we obtain the characteristic curve of the bias-variance trade-off. This suggests that the quantity of information in the weights correlates well with generalization.

nuisance. Recall that by definition we have

$$\begin{aligned} I(z;n) &= \mathbb{E}_{n\sim p(n)} \mathrm{KL}(p(z|n) \parallel p(z)) \\ &= \mathbb{E}_{n\sim p(n)} \mathbb{E}_{z\sim p(z|n)} \log[p(z|n)/p(z)]. \end{aligned}$$

To approximate the expectations via sampling we need a way to approximate the likelihood ratio $\log p(\mathbf{z}|\mathbf{n})/p(\mathbf{z})$. This can be done as follows: Let $D(z;n)$ be a binary discriminator that given the representation z and the nuisance n tries to decide whether z is sampled from the posterior distribution $p(z|n)$ or from the prior $p(z)$. Since by hypothesis we can generate samples from both distributions, we can generate data to train this discriminator. Intuitively, if the discriminator is not able to classify, it means that z is insensitive to changes of n. Precisely, since the optimal discriminator is

$$D^*(z;n) = \frac{p(z)}{p(z) + p(z|n)},$$

if we assume that D is close to the optimal discriminator D^*, we have

$$\log \frac{p(z|n)}{p(z)} = \log \frac{1 - D^*(z;n)}{D^*(z;n)} \simeq \log \frac{1 - D(z;n)}{D(z;n)}.$$

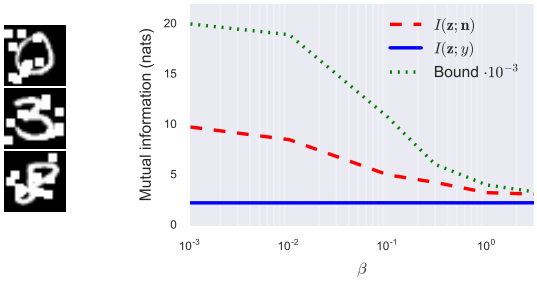


Figure 4.5: **(Left)** A few training samples generated adding nuisance clutter n to the MNIST dataset. **(Right)** Reducing the information in the weights makes the representation z learned by the digit classifier increasingly invariant to nuisances ($I(n;z)$ decreases), while sufficiency is retained ($I(z;y) = I(x;y)$ is constant). As expected, $I(z;n)$ is smaller but has a similar behavior to the theoretical bound in Corollary 4.5.3.

therefore we can use D to estimate the log-likelihood ratio, and so also the mutual information $I(z;n)$. Notice however that this comes with no guarantees on the quality of the approximation.

To test this algorithm, we add random occlusion nuisances to MNIST digits (Figure 4.5). In this case, the nuisance n is the occlusion pattern, while the observed data x is the occluded digit. For various values of β, we train a classifier on this data in order to learn a representation z, and, for each representation obtained this way, we train a discriminator as described above and we compute the resulting approximation of $I(z;n)$. The results in Figure 4.5 show that decreasing the information in the weights makes the representation increasingly more insensitive to n.

4.7 Discussion and conclusion

In this chapter, we have presented bounds, some of which are tight, that connect the amount of information in the weights, the amount of information in the activations, the invariance property of the network, and the geometry of the residual loss. These results leverage the structure of deep networks, in particular the multiplicative action of the weights, and the Markov property of the layers. This leads to the surprising result that reducing information

stored in the weights about the past (dataset) results in desirable properties of the learned internal representation of the test datum (future).

Our notion of representation, both the weights and the activations, is intrinsically stochastic. This simplifies the computation as well as the derivation of information-based relations. However, note that even if we start with a deterministic representation w, Proposition 4.2.2 gives us a way of converting it to a stochastic representation whose quality depends on the flatness of the minimum. Our theory can be derived from different frameworks, such as MDL, PAC-Bayes and Information Theory, and implies a novel variant of the Information Bottleneck Principle which applies to the weights of the network, rather than the activations.

This work focuses on the inference and learning of optimal representations, that seek to get the most out of the data we have for a specific task. This does not guarantee a good outcome since, due to the Data Processing Inequality, the representation can be easier to use but ultimately no more informative than the data themselves. An orthogonal but equally interesting issue is how to get the most informative data possible, which is the subject of active learning, experiment design, and perceptual exploration. Our work does not address transfer learning, where a representation trained to be optimal for a task is instead used for a different task, which will be subject of future investigations.

4.8 Related work

The Information Bottleneck (IB) was introduced by Tishby et al. (1999) as a generalization of minimal sufficient statistics that allows trading off fidelity (sufficiency) and complexity of a representation. In particular, the IB Lagrangian reduces finding a minimal sufficient representation to a variational optimization problem. Later, Tishby and Zaslavsky (2015) and Shwartz-Ziv and Tishby (2017) advocated using the IB between the test data and the activations of a deep neural network, to study the sufficiency and minimality of the resulting representation. In parallel developments, the IB Lagrangian was used as a regularized loss function for learning representation, leading to new information theoretic regularizers (see also Chapter 2).

In this chapter, we introduce an IB Lagrangian between the weights of a network and the training data, as opposed to the traditional one between the activations and the test datum. We show that the former can be seen both as a generalization of Variational Inference, related to Hinton and Van Camp (1993), and as a special case of the more general PAC-Bayes framework (McAllester, 2013), that can be used to compute high-probability upper-bounds on the test error of the network. One of our main contributions is then to show that, due to a particular duality induced by the architecture of deep networks, minimality of the weights (a function of the training dataset) and of the learned representation (a function of the test input) are connected: in particular we show that networks regularized either explicitly, or implicitly by SGD, are biased toward learning invariant and disentangled representations. The theory we develop could be used to explain the phenomena described in small-scale experiments in Shwartz-Ziv and Tishby (2017), whereby the initial fast convergence of SGD is related to sufficiency of the representation, while the later asymptotic phase is related to compression of the activations: While SGD is seemingly agnostic to the property of the learned representation, we show that it does minimize the information in the weights, from which the compression of the activations follows as a corollary of our bounds. Practical implementation of this theory on real large scale problems is made possible by advances in Stochastic Gradient Variational Bayes (Kingma and Welling, 2014; Kingma et al., 2015).

Representations learned by deep networks are observed to be insensitive to complex nuisance transformations of the data. To a certain extent, this can be attributed to the architecture. For instance, the use of convolutional layers and max-pooling can be shown to yield insensitivity to local group transformations (Bruna and Mallat, 2011; Anselmi et al., 2016; Soatto and Chiuso, 2016). But for more complex, dataset-specific, and in particular non-local, non-group transformations, such insensitivity must be acquired as part of the learning process, rather than being coded in the architecture. We show that a sufficient representation is maximally insensitive to nuisances if and only if it is minimal, allowing us to prove that a regularized network is naturally biased toward learning invariant representations of the data.

We have also explored relations between our theory and the PAC-Bayes framework. The

use of PAC-Bayes theory to study the generalization properties of deep networks has been championed by Dziugaite and Roy (2017), who point out that minima that are flat in the sense of having a large volume, toward which stochastic gradient descent algorithms are implicitly or explicitly biased (Chaudhari and Soatto, 2018), naturally relates to the PAC-Bayes loss for the choice of a normal prior and posterior on the weights. This has been leveraged by Dziugaite and Roy (2017) to compute non-vacuous PAC-Bayes error bounds, even for deep networks.

CHAPTER 5

Asymmetric distance between tasks

Among the many virtues of deep neural networks is their *transferability*: One can train a model for a task (*e.g.,* finding cats and dogs in images), and then use it for another (*e.g.,* outlining tumors in mammograms) with relatively little effort. However, little is known on how to predict whether or not such *transfer learning* will work, and if so how much effort is going to be needed, without just trying-and-seeing. It is not a given that training on a sufficiently rich task, and then fine-tuning on anything else, must succeed. Indeed, slight changes in the statistics of the data can make a task *unreachable*, as we will see in Chapter 6

At the most fundamental level, understanding transfer learning or domain adaptation requires understanding the *topology and geometry of the space of tasks.* When are two tasks "close"? Can one measure the distance between tasks without actually running an experiment? Does knowing this distance help predict whether transfer learning is possible, and if so how many resources or time will be needed?

Surely the distance between tasks is not just the lexicographic distance between label sets in a taxonomy: The experiments we will present in Chapter 6 show that even for the same label set, a task can be unreachable. Surely it is also not just the distance between two sets of parameters in a model (say, a deep neural network) trained for the task: There are many symmetries and large subsets in parameter space that implement the same model. These questions are fundamental because they do not concern a particular choice of model (*e.g.,* neural networks) or optimization scheme (*e.g.,* stochastic gradient descent, SGD). They are questions about *learnability* of tasks, and *transferability* from a task to another. But what is a *task*? What is its complexity? What is its structure?

In this chapter, we introduce a method to provide vectorial representations of visual

classification tasks which can be used to reason about the nature of those tasks and their distance. Given a dataset with ground-truth labels and a loss function defined over those labels, we process images through a "probe network" and compute an embedding based on estimates of the Fisher information matrix associated with the probe network parameters. This provides a fixed-dimensional embedding of the task that is independent of details such as the number of classes and does not require any understanding of the class label semantics. We demonstrate that this embedding is capable of predicting task similarities that match our intuition about semantic and taxonomic relations between different visual tasks (*e.g.*, tasks based on classifying different types of plants are similar). We also demonstrate the practical value of this framework for the meta-task of selecting a pre-trained feature extractor for a new task. We present a simple meta-learning framework for learning a metric on embeddings that is capable of predicting which feature extractors will perform well. Selecting a feature extractor with task embedding obtains a performance close to the best available feature extractor, while costing substantially less than exhaustively training and evaluating on all available feature extractors.

5.1 Task Embeddings via Fisher Information

Given an observed input x (*e.g.*, an image) and an hidden task variable y (*e.g.*, a label), a deep network is a family of functions $p_w(y|x)$ parametrized by weights w, trained to approximate the posterior $p(y|x)$ by minimizing the (possibly regularized) cross entropy loss $H_{p_w,\hat{p}}(y|x) = \mathbb{E}_{x,y\sim\hat{p}}[-\log p_w(y|x)]$, where $\hat{p}$ is the empirical distribution defined by the training set $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$. It is useful, especially in transfer learning, to think of the network as composed of two parts: a feature extractor which computes some representation $z = \phi_w(x)$ of the input data, and a "head," or classifier, which encodes the distribution $p(y|z)$ given the representation z.

Not all network weights are equally useful in predicting the task variable: the importance, or "informative content," of a weight for the task can be quantified by considering a perturbation $w' = w + \delta w$ of the weights, and measuring the average KL divergence between

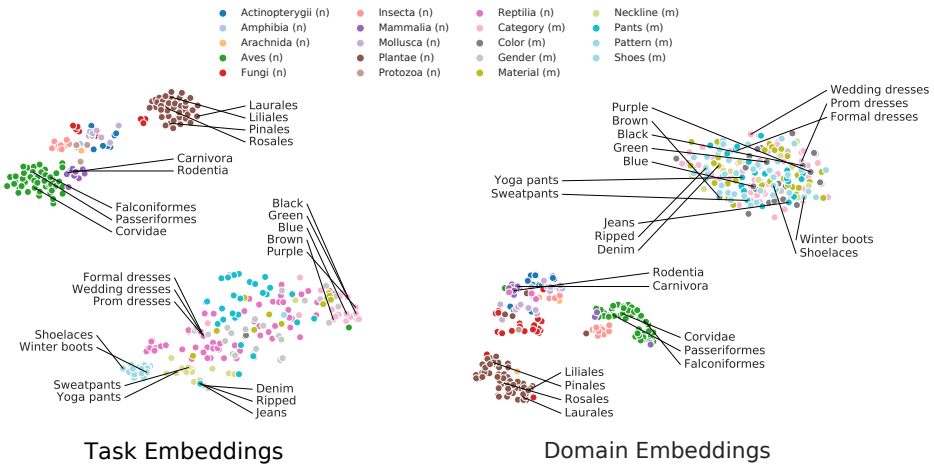


Figure 5.1: **Task embedding across a large library of tasks** (best seen magnified). **(Left)** T-SNE visualization of the embedding of tasks extracted from the iNaturalist, CUB-200, iMaterialist datasets. Colors indicate ground-truth grouping of tasks based on taxonomic or semantic types. Notice that the bird classification tasks extracted from CUB-200 embed near the bird classification task from iNaturalist, even though the original datasets are different. iMaterialist is well separated from iNaturalist, as it entails very different tasks (clothing attributes). Notice that some tasks of similar type (such as color attributes) cluster together but attributes of different task types may also mix when the underlying visual semantics are correlated. For example, the tasks of jeans (clothing type), denim (material) and ripped (style) recognition are close in the task embedding. **(Right)** T-SNE visualization of the domain embeddings (using mean feature activations) for the same tasks. Domain embedding can distinguish iNaturalist tasks from iMaterialist tasks due to differences in the two problem domains. However, the fashion attribute tasks on iMaterialist all share the same domain and only differ in their labels. In this case, the domain embeddings collapse to a region without recovering any sensible structure.

the original output distribution $p_w(y|x)$ and the perturbed one $p_{w'}(y|x)$. To second-order approximation, this is

$$\mathbb{E}_{x\sim\hat{p}}\,\mathrm{KL}(p_{w'}(y|x)\parallel p_w(y|x)) = \delta w\cdot F\delta w + o(\delta w^2),$$

where F is the Fisher information matrix (FIM):

$$F = \mathbb{E}_{x,y\sim\hat{p}(x)p_w(y|x)}\left[\nabla_w \log p_w(y|x) \nabla_w \log p_w(y|x)^T\right],$$

that is, the expected covariance of the scores (gradients of the log-likelihood) with respect to the model parameters.

The FIM is a Riemannian metric on the space of probability distributions (Amari and Nagaoka, 2000a), and provides a measure of the information a particular parameter (weight or feature) contains about the joint distribution $p_w(x, y) = \hat{p}(x)p_w(y|x)$: If the classification performance for a given task does not depend strongly a parameter, the corresponding entry in the FIM will be small. Notice that we have already seen the Fisher Information Matrix in Proposition 4.2.2, where we interpreted (its log-determinant) as a measure of the information needed to solve the task at a certain level, estimated using an uninfomrative prior over the solutions. Moreover, the FIM can be interpreted as an easy-to-compute positive semidefinite upper-bound to the Hessian of the cross-entropy loss, and coincides with it at local minima (Martens, 2014). In particular, "flat minima" correspond to weights that have, on average, low (Fisher) information (Remark 4.2.3).

TASK2VEC embedding using a probe network

While the network activations capture the information in the input image which are needed to infer the image label, the FIM indicates the set of feature maps which are more informative for solving the current task. Following this intuition, we use the FIM to represent the task itself. However, the FIMs computed on different networks are not directly comparable. To address this, we use single "probe" network pre-trained on ImageNet as a feature extractor and re-train only the classifier layer on any given task, which usually can be done efficiently. After training is complete, we compute the FIM for the feature extractor parameters.

Since the full FIM is unmanageably large for rich probe networks based on CNNs, we make two additional approximations. First, we only consider the diagonal entries, which implicitly assumes that correlations between different filters in the probe network are not important. Second, since the weights in each filter are usually not independent, we average

the Fisher Information for all weights in the same filter. The resulting representation thus has fixed size, equal to the number of filters in the probe network. We call this embedding method TASK2VEC.

Robust Fisher computation Since the FIM is a local quantity, it is affected by the local geometry of the training loss landscape, which is highly irregular in many deep network architectures (Li et al., 2017), and may be too noisy when trained with few samples. To avoid this problem, instead of a direct computation, we use a more robust estimator that leverages connections to variational inference. Assume we perturb the weights $\hat{w}$ of the network with Gaussian noise $\mathcal{N}(0, \Lambda)$ with precision matrix Λ, and we want to find the optimal Λ which yields a good expected error, while remaining close to an isotropic prior $\mathcal{N}(\hat{w}, \lambda^2 I)$. That is, we want to find Λ that minimizes:

$$L(\hat{w}; \Lambda) = \mathbb{E}_{w \sim \mathcal{N}(\hat{w}, \Lambda)}[H_{p_w, \hat{p}} p(y|x)] + \beta \, \mathrm{KL}(\mathcal{N}(0, \Lambda) \parallel \mathcal{N}(0, \lambda^2 I)),$$

where H is the cross-entropy loss and β controls the weight of the prior. Notice that for $\beta = 1$ this reduces to the Evidence Lower-Bound (ELBO) commonly used in variational inference. Approximating to the second order, the optimal value of Λ satisfies (Proposition 4.2.2):

$$\frac{\beta}{2N} \Lambda = F + \frac{\beta \lambda^2}{2N} I.$$

Therefore, $\frac{\beta}{2N} \Lambda \sim F + o(1)$ can be considered as an estimator of the FIM F, biased towards the prior $\lambda^2 I$ in the low-data regime instead of being degenerate. In case the task is trivial (the loss is constant or there are too few samples) the embedding will coincide with the prior $\lambda^2 I$, which we will refer to as the **trivial embedding**. This estimator has the advantage of being easy to compute by directly minimizing the loss $L(\hat{w}; \Sigma)$ through Stochastic Gradient Variational Bayes (Kingma et al., 2015), while being less sensitive to irregularities of the loss landscape than direct computation, since the value of the loss depends on the cross-entropy in a neighborhood of $\hat{w}$ of size Λ^{-1}. As in the standard Fisher computation, we estimate one parameter per filter, rather than per weight, which in practice means that we constrain $\Lambda_{ii} = \Lambda_{jj}$ whenever w_i and w_j belongs to the same filter. In this case, optimization of $L(\hat{w}; \Lambda)$ can be done efficiently using the local reparametrization trick of Kingma et al. (2015).

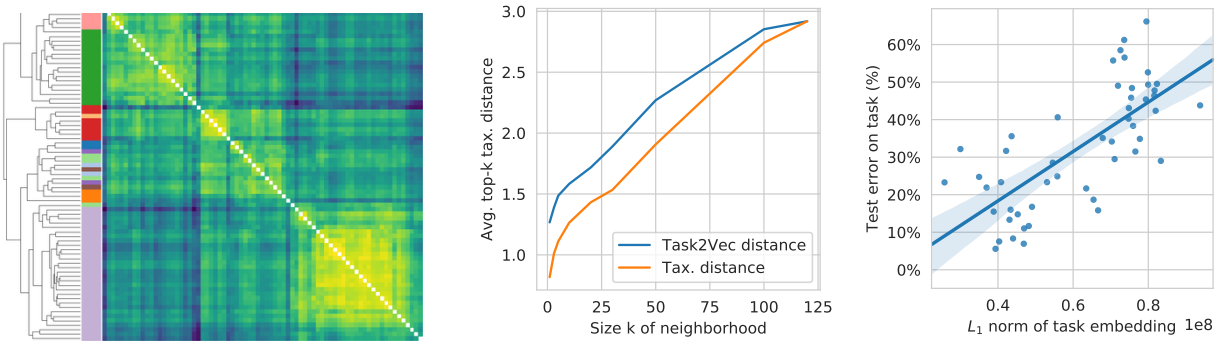


Figure 5.2: **Distance between species classification tasks.** **(Left)** Task similarity matrix ordered by hierarchical clustering. Note that the dendrogram produced by the task similarity matches the taxonomic clusters (indicated by color bar). **(Center)** For tasks extracted from iNaturalist and CUB, we compare the cosine distance between tasks to their taxonomical distance. As the size of the task embedding neighborhood increases (measured by number of tasks in the neighborhood), we plot the average taxonomical distance of tasks from the neighborhood center. While the task distance does not perfectly match the taxonomical distance (whose curve is shown in orange), it shows a good correlation. Difference are both due to the fact that taxonomically close species may need very different features to be classified, creating a mismatch between the two notions of distance, and because for some tasks in iNaturalist too few samples are provided to compute a good embedding. **(Right)** Correlation between L_1 norm of the task embedding (distance from origin) and test error obtained on the task.

Properties of the TASK2VEC embedding

The task embedding we just defined has a number of useful properties. For illustrative purposes, consider a two-layer sigmoidal network for which an analytic expression can be derived (see Supplementary Materials). The FIM of the feature extractor parameters can be written using the Kronecker product as

$$F = \mathbb{E}_{x,y\sim\hat{p}(x)p_w(y|x)}[(y-p)^2 \cdot S \otimes xx^T]$$

where $p = p_w(y=1|x)$ and the matrix $S = ww^T \odot zz^T \odot (1-z)(1-z)^T$ is an element-wise product of classifier weights w and first layer feature activations z. It is informative

to compare this expression to an embedding based only on the dataset domain statistics, such as the (non-centered) covariance $C_0 = \mathbb{E}\left[xx^T\right]$ of the input data or the covariance $C_1 = \mathbb{E}\left[zz^T\right]$ of the feature activations. One could take such statistics as a representative *domain embedding* since they only depend on the marginal distribution $p(x)$ in contrast to the FIM *task embedding*, which depends on the joint distribution $p(x, y)$. These simple expressions highlight some important (and more general) properties of the Fisher embedding we now describe.

Invariance to the label space: The task embedding does not directly depend on the task labels, but only on the predicted distribution $p_w(y|x)$ of the trained model. Information about the ground-truth labels y is encoded in the weights w which are a sufficient statistic of the task (Chapter 4). In particular, the task embedding is invariant to permutations of the labels y, and has fixed dimension (number of filters of the feature extractor) regardless of the output space (e.g., k-way classification with varying k).

Encoding task difficulty: As we can see from the expressions above, if the fit model is very confident in its predictions, $\mathbb{E}[(y-p)^2]$ goes to zero. Hence, the norm of the task embedding $\|F\|_\star$ scales with the difficulty of the task for a given feature extractor ϕ. Figure 5.2 (Right) shows that even for more complex models trained on real data, the FIM norm correlates with test performance.

Encoding task domain: Data points x that are classified with high confidence, i.e., p is close to 0 or 1, will have a lower contribution to the task embedding than points near the decision boundary since $p(1-p)$ is maximized at $p = 1/2$. Compare this to the covariance matrix of the data, C_0, to which all data points contribute equally. Instead, in TASK2VEC information on the domain is based on data near the decision boundary (task-weighted domain embedding).

Encoding useful features for the task: The FIM depends on the curvature of the loss function with the diagonal entries capturing the sensitivity of the loss to model parameters. Specifically, in the two-layer model one can see that, if a given feature is uncorrelated with y, the corresponding blocks of F are zero. In contrast, a domain embedding based on feature

activations of the probe network (e.g., C_1) only reflects which features vary over the dataset without indication of whether they are relevant to the task.

5.2 Similarity Measures on the Space of Tasks

What metric should be used on the space of tasks? This depends critically on the meta-task we are considering. As a motivation, we concentrate on the meta-task of selecting the pre-trained feature extractor from a set in order to obtain the best performance on a new training task. There are several natural metrics that may be considered for this meta-task. In this work, we mainly consider:

Taxonomic distance For some tasks, there is a natural notion of semantic similarity, for instance defined by sets of categories organized in a taxonomic hierarchy where each task is classification inside a subtree of the hierarchy (*e.g.*, we may say that classifying breeds of dogs is closer to classification of cats than it is to classification of species of plants). In this setting, we can define

$$D_{\text{tax}}(t_a, t_b) = \min_{i \in S_a, j \in S_b} d(i, j),$$

where S_a, S_b are the sets of categories in task t_a, t_b and $d(i, j)$ is an ultrametric or graph distance in the taxonomy tree. Notice that this is a proper distance, and in particular it is symmetric.

Transfer distance. We define the transfer (or fine-tuning) gain from a task t_a to a task t_b (which we improperly call distance, but is not necessarily symmetric or positive) as the difference in expected performance between a model trained for task t_b from a fixed initialization (random or pre-trained), and the performance of a model fine-tuned for task t_b starting from a solution of task t_a:

$$D_{\text{ft}}(t_a \to t_b) = \frac{\mathbb{E}[\ell_{a \to b}] - \mathbb{E}[\ell_b]}{\mathbb{E}[\ell_b]},$$

where the expectations are taken over all trainings with the selected architecture, training procedure and network initialization, ℓ_b is the final test error obtained by training on task b

from the chosen initialization, and $\ell_{a\to b}$ is the error obtained instead when starting from a solution to task a and then fine-tuning (with the selected procedure) on task t_b.

Symmetric and asymmetric TASK2VEC metrics

By construction, the Fisher embedding on which TASK2VEC is based captures fundamental information about the structure of the task. We may therefore expect that the distance between two embeddings correlate positively with natural metrics on the space of tasks. However, there are two problems in using the Euclidean distance between embeddings: the parameters of the network have different scales, and the norm of the embedding is affected by complexity of the task and the number of samples used to compute the embedding.

Symmetric TASK2VEC distance To make the distance computation robust, we propose to use the cosine distance between normalized embeddings:

$$d_{\text{sym}}(F_a, F_b) = d_{\cos}\Big(\frac{F_a}{F_a + F_b}, \frac{F_b}{F_a + F_b}\Big),$$

where $d_{\cos}$ is the cosine distance, F_a and F_b are the two task embeddings (*i.e.*, the diagonal of the Fisher Information computed on the same probe network), and the division is element-wise. This is a symmetric distance which we expect to capture semantic similarity between two tasks. For example, we show in Fig. 5.2 that it correlates well with the taxonomical distance between species on iNaturalist.

On the other hand, precisely for this reason, this distance is ill-suited for tasks such as model selection, where the (intrinsically asymmetric) transfer distance is more relevant.

Asymmetric TASK2VEC distance In a first approximation, that does not consider either the model or the training procedure used, positive transfer between two tasks depends both on the similarity between two tasks and on the complexity of the first. Indeed, pre-training on a general but complex task such as ImageNet often yields a better result than fine-tuning from a close dataset of comparable complexity. In our case, complexity can be measured as the distance from the trivial embedding. This suggests the following asymmetric score,

again improperly called a "distance" despite being asymmetric and possibly negative:

$$d_{\text{asym}}(t_a \to t_b) = d_{\text{sym}}(t_a, t_b) - \alpha d_{\text{sym}}(t_a, t_0),$$

where t_0 is the trivial embedding, and α is an hyperparameter. This has the effect of bring more complex models closer. The hyper-parameter α can be selected based on the meta-task. In our experiments, we found that the best value of α ($\alpha = 0.15$ when using a ResNet-34 pretrained on ImageNet as the probe network) is robust to the choice of meta-tasks.

5.3 MODEL2VEC: task/model co-embedding

By construction, the TASK2VEC distance ignores details of the model and only relies on the task. If we know what task a model was trained on, we can represent the model by the embedding of that task. However, in general we may not have such information (*e.g.*, black-box models or hand-constructed feature extractors). We may also have multiple models trained on the same task with different performance characteristics. To model the joint interaction between task and model (*i.e.*, architecture and training algorithm), we aim to learn a joint embedding of the two.

We consider for concreteness the problem of learning a joint embedding for model selection. In order to embed models in the task space so that those near a task are likely to perform well on that task, we formulate the following meta-learning problem: Given k models, their MODEL2VEC embedding are the vectors $m_i = F_i + b_i$, where F_i is the task embedding of the task used to train model m_i (if available, else we set it to zero), and b_i is a learned "model bias" that perturbs the task embedding to account for particularities of the model. We learn b_i by optimizing a k-way cross entropy loss to predict the best model given the task distance (see Supplementary Material):

$$\mathcal{L} = \mathbb{E}[-\log p(m \,|\, d_{\text{asym}}(t, m_0), \ldots, d_{\text{asym}}(t, m_k))].$$

After training, given a novel query task t, we can then predict the best model for it as the $\arg\max_i d_{\text{asym}}(t, m_i)$, that is, the model m_i embedded closest to the query task.

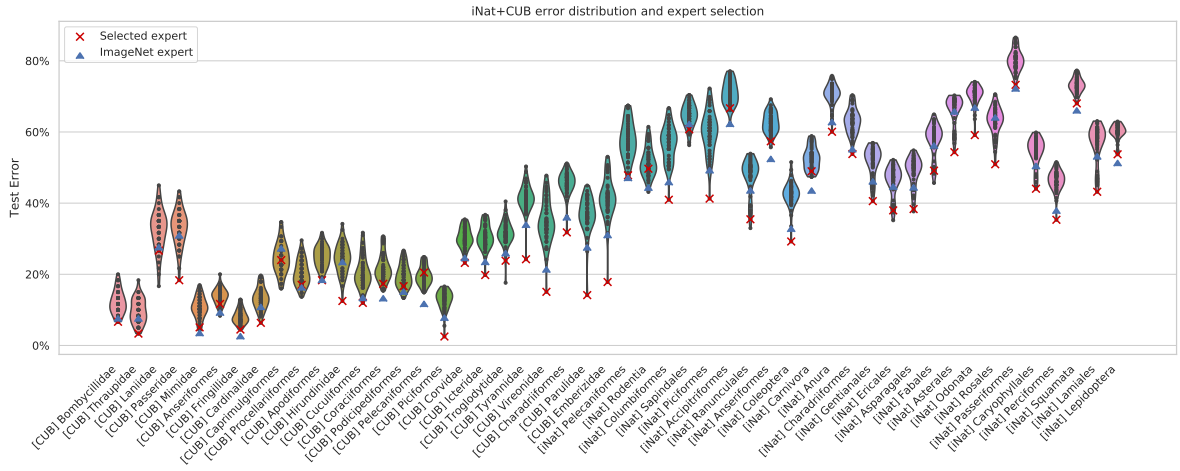


Figure 5.3: **TASK2VEC often selects the best available experts.** Violin plot of the distribution of the final test error (shaded plot) on tasks from the CUB-200 dataset (columns) obtained by training a linear classifier over several expert feature extractors (points). Most specialized feature extractors perform similarly on a given task, and generally are similar or worse than a generic feature extractor pre-trained on ImageNet (blue triangles). However, in some cases a carefully chosen expert, trained on a relevant task, can greatly outperform all other experts (long whisker of the violin plot). The model selection algorithm based on TASK2VEC can, without training, suggest an expert to use for the task (red cross, lower is better). TASK2VEC mostly recover the optimal, or close to optimal, feature extractor to use without having to perform an expensive brute-force search over all possibilities. Columns are ordered by norm of the task embedding: Notice tasks with lower embedding norm have lower error and more "complex" task (task with higher embedding norm) tend to benefit more from a specialized expert.

5.4 Experiments

We test TASK2VEC on a large collection of tasks and models, related to different degrees. Our experiments aim to test both qualitative properties of the embedding and its performance on meta-learning tasks. We use an off-the-shelf ResNet-34 pretrained on ImageNet as our probe network, which we found to give the best overall performance (see Sect. 5.4). The collection of tasks is generated starting from the following four main datasets. **iNaturalist** Van Horn et al. (2018): Each task extracted corresponds to species classification in a given taxonomical order. For instance, the *"Rodentia task"* is to classify species of rodents. Notice that each task is defined on a separate subset of the images in the original dataset; that is, the domains of the tasks are disjoint. **CUB-200** Wah et al. (2011): We use the same procedure as iNaturalist to create tasks. In this case, all tasks are classifications inside orders of birds (the *aves* taxonomical class), and have generally much less training samples than corresponding tasks in iNaturalist. **iMaterialist** iMa and **DeepFashion** Liu et al. (2016): Each image in both datasets is associated with several binary attributes (*e.g.*, style attributes) and categorical attributes (*e.g.*, color, type of dress, material). We binarize the categorical attributes, and consider each attribute as a separate task. Notice that, in this case, all tasks share the same domain and are naturally correlated.

In total, our collection of tasks has 1460 tasks (207 iNaturalist, 25 CUB, 228 iMaterialist, 1000 DeepFashion). While a few tasks have many training examples (*e.g.*, hundred thousands), most have just hundreds or thousands of samples. This simulates the heavy-tail distribution of data in real-world applications.

Together with the collection of tasks, we collect several "expert" feature extractors. These are ResNet-34 models pre-trained on ImageNet and then fine-tuned on a specific task or collection of related tasks (see Supplementary Materials for details). We also consider a "generic"expert pre-trained on ImageNet without any finetuning. Finally, for each combination of expert feature extractor and task, we trained a linear classifier on top of the expert in order to solve the selected task using the expert.

In total, we trained 4,100 classifiers, 156 feature extractors and 1,460 embeddings. The

total effort to generate the final results was about 1,300 GPU hours.

Meta-tasks. In Section 5.4, for a given task we aim to predict, using TASK2VEC , which expert feature extractor will yield the best classification performance. In particular, we formulate two model selection meta-tasks: **iNat + CUB** and **Mixed**. The first consists of 50 tasks and experts from iNaturalist and CUB, and aims to test fine-grained expert selection in a restricted domain. The second contains a mix of 26 curated experts and 50 random tasks extracted from all datasets, and aims to test model selection between different domains and tasks.

Task Embedding Results

Task Embedding qualitatively reflects taxonomic distance for iNaturalist For tasks extracted from the iNaturalist dataset (classification of species), the taxonomical distance between orders provides a natural metric of the semantic similarity between tasks. In Figure 5.2 we compare the symmetric TASK2VEC distance with the taxonomical distance, showing strong agreement.

Task embedding for iMaterialist In Fig. 5.1 we show a t-SNE visualization of the embedding for iMaterialist and iNaturalist tasks. Task embedding yields interpretable results: Tasks that are correlated in the dataset, such as binary classes corresponding to the same categorical attribute, may end up far away from each other and close to other tasks that are semantically more similar (*e.g.*, the *jeans* category task is close to the *ripped* attribute and the *denim* material). This is reflected in the mixture of colors of semantically related nearby tasks, showing non-trivial grouping.

We also compare the TASK2VEC embedding with a domain embedding baseline, which only exploits the input distribution $p(x)$ rather than the task distribution $p(x, y)$. While some tasks are highly correlated with their domain (*e.g.*, tasks from iNaturalist), other tasks differ only on the labels (*e.g.*, all the attribute tasks of iMaterialist, which share the same clothes domain). Accordingly, the domain embedding recovers similar clusters on iNaturalist.

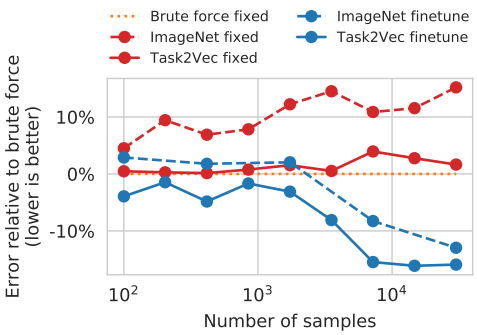


Figure 5.4: **TASK2VEC improves results at different dataset sizes and training conditions:** Performance of model selection on a subset of 4 tasks as a function of the number of samples available to train relative to optimal model selection (dashed orange). Training a classifier on the feature extractor selected by TASK2VEC (solid red) is always better than using a generic ImageNet feature extractor (dashed red). The same holds when allowed to fine-tune the feature extractor (blue curves). Also notice that in the low-data regime fine-tuning the ImageNet feature extractor is more expensive and has a worse performance than accurately selecting a good fixed feature extractor.

However, on iMaterialst domain embedding collapses all tasks to a single uninformative cluster (not a single point due to slight noise in embedding computation).

Task Embedding encodes task difficulty The scatter-plot in Fig. 5.3 compares the norm of embedding vectors vs. performance of the best expert (or task specific model for cases where we have the diagonal computed). As shown analytically for the two-layers model, the norm of the task embedding correlates with the complexity of the task also on real tasks and architectures.

Model Selection

Given a task, our aim is to select an expert feature extractor that maximizes the classification performance on that task. We propose two strategies: (1) embed the task and select the feature extractor trained on the most similar task, and (2) jointly embed the models and

Probe network	Top-10	All
Chance	+13.95%	+59.52%
VGG-13	+4.82%	+38.03%
DenseNet-121	+0.30%	+10.63%
ResNet-13	**+0.00%**	**+9.97%**

Table 5.1: **Choice of probe network.** Mean relative error increase over the ground-truth optimum on the iNat+CUB meta-task for different choices of the probe-network. We also report the performance on the top 10 tasks with more samples to show how data size affect different architectures.

tasks, and select a model using the learned metric (see Section 5.3). Notice that (1) does not use knowledge of the model performance on various tasks, which makes it more widely applicable but requires we know what task a model was trained for and may ignore the fact that models trained on slightly different tasks may still provide an overall better feature extractor (for example by over-fitting less to the task they were trained on).

In Table 5.2 we compare the overall results of the various proposed metrics on the model selection meta-tasks. On both the iNat+CUB and Mixed meta-tasks, the Asymmetric TASK2VEC model selection is close to the ground-truth optimal, and significantly improves over both chance, and over using an generic ImageNet expert. Notice that our method has $O(1)$ complexity, while searching over a collection of N experts is $O(N)$.

Error distribution In Fig. 5.3 we show in detail the error distribution of the experts on multiple tasks. It is interesting to notice that the classification error obtained using most experts clusters around some mean value, and little improvement is observed over using a generic expert. On the other hand, a few optimal experts can obtain a largely better performance on the task than a generic expert. This confirms the importance of having access to a large collection of experts when solving a new task, especially if few training data are available. But this collection can only be efficiently exploited if an algorithm is given to efficiently find one of the few experts for the task, which we propose.

Meta-task	Optimal	Chance	ImageNet	TASK2VEC	Asym. TASK2VEC	MODEL2VEC
iNat + CUB	31.24	+59.52%	+30.18%	+42.54%	+9.97%	**+6.81%**
Mixed	22.90	+112.49%	+75.73%	+40.30%	+29.23%	**+27.81%**

Table 5.2: **Model selection performance of different metrics.** Average optimal error obtained on two meta-learning tasks by exhaustive search over the best expert, and relative error increase when using cheaper model selection methods. Always picking a fixed good general model (*e.g.*, a model pretrained on ImageNet) performs better than picking an expert at random (chance). However, picking an expert using the Asymmetric TASK2VEC distance can achieve an overall better performance than using a general model. Notice also the improvement over the Symmetric version, especially on iNat + CUB, where experts trained on very similar tasks may be too simple to yield good transfer, and should be avoided.

Dependence on task dataset size Finding experts is especially important when the task we are interested in has relatively few samples. In Fig. 5.4 we show how the performance of TASK2VEC varies on a model selection task as the number of samples varies. At all sample sizes TASK2VEC is close to the optimum, and improves over selecting a generic expert (ImageNet), both when fine-tuning and when training only a classifier. We observe that the best choice of experts is not affected by the dataset size, and that even with few examples TASK2VEC is able to find the optimal experts.

Choice of probe network In Table 5.1 we show that DenseNet (Huang et al., 2017) and ResNet architectures (He et al., 2016) perform significantly better when used as probe networks to compute the TASK2VEC embedding than a VGG Simonyan and Zisserman (2014) architecture.

5.5 Related Work

Task and Domain embedding. Tasks distinguished by their domain can be understood simply in terms of image statistics. Due to the bias of different datasets, sometimes a

benchmark task may be identified just by looking at a few images (Torralba and Efros, 2011). The question of determining what summary statistics are useful (analogous to our choice of probe network) has also been considered, for example Edwards and Storkey (2016) train an autoencoder that learns to extract fixed dimensional summary statistics that can reproduce many different datasets accurately. However, for general vision tasks which apply to all natural images, the domain is the same across tasks.

Taskonomy (Zamir et al., 2018) explores the structure of the space of tasks, focusing on the question of effective knowledge transfer in a curated collection of 26 visual tasks, ranging from classification to 3D reconstruction, defined on a common domain. They compute pairwise transfer distances between pairs of tasks and use the results to compute a directed hierarchy. Introducing novel tasks requires computing the pairwise distance with tasks in the library. In contrast, we focus on a larger library of 1,460 fine-grained classification tasks both on same and different domains, and show that it is possible to represent tasks in a topological space with a constant-time embedding. The large task collection and cheap embedding costs allow us to tackle new meta-learning problems.

Fisher kernels Our work takes inspiration from Jaakkola and Hausler (Jaakkola and Haussler, 1999). They propose the "Fisher Kernel", which uses the gradients of a generative model score function as a representation of similarity between data items

$$K(x^{(1)}, x^{(2)}) = \nabla_\theta \log P(x^{(1)}|\theta)^T F^{-1} \nabla_\theta \log P(x^{(2)}|\theta).$$

Here $P(x|\theta)$ is a parameterized generative model and F is the Fisher information matrix. This provides a way to utilize generative models in the context of discriminative learning. Variants of the Fisher kernel have found wide use as a representation of images (Perronnin et al., 2010; Sánchez et al., 2013), and other structured data such as protein molecules (Jaakkola et al., 1999) and text (Saunders et al., 2003). Since the generative model can be learned on unlabelled data, several works have investigated the use of Fisher kernel for unsupervised learning (Holub et al., 2005; Seeger, 2000). Van Der Maaten (2011) learns a metric on the Fisher kernel representation similar to our metric learning approach. Our

approach differs in that we use the FIM as a representation of a whole dataset (task) rather than using model gradients as representations of individual data items.

Fisher Information for CNNs Our approach to task embedding makes use of the Fisher Information matrix of a neural network as a characterization of the task. Use of Fisher information for neural networks was popularized by Amari Amari (1998) who advocated optimization using natural gradient descent which leverages the fact that the FIM is an appropriate parameterization-independent metric on statistical models. Recent work has focused on approximates of FIM appropriate in this setting (see e.g., Heskes (2000); Finn et al. (2017); Martens and Grosse (2015)). FIM has also been proposed for various regularization schemes (see Chapter 4, Arora et al. (2018); Liang et al. (2017); Mroueh and Sercu (2017)), analyze learning dynamics of deep networks Achille et al. (2019), and to overcome catastrophic forgetting (Kirkpatrick et al., 2017b).

Meta-learning and Model Selection The general problem of meta-learning has a long history with much recent work dedicated to problems such as neural architecture search and hyper-parameter estimation. Closely related to our problem is work on selecting from a library of classifiers to solve a new task (Smith et al., 2014; Abdulrahman et al., 2018; Leite et al., 2012). Unlike our approach, these usually address the question via land-marking or active testing, in which a few different models are evaluated and performance of the remainder estimated by extension. This can be viewed as a problem of completing a matrix defined by performance of each model on each task.

A similar approach has been taken in computer vision for selecting a detector for a new category out of a large library of detectors (Matikainen et al., 2012; Zhang et al., 2014; Wang and Hebert, 2015).

5.6 Discussion

TASK2VEC is an efficient way to represent a task, or the corresponding dataset, as a fixed dimensional vector. It has several appealing properties, in particular its norm correlates with the test error obtained on the task, and the cosine distance between embeddings correlates with natural distances between tasks, when available, such as the taxonomic distance for species classification, and the fine-tuning distance for transfer learning. Having a representation of tasks paves the way for a wide variety of meta-learning tasks. In this work, we focused on selection of an expert feature extractor in order to solve a new task, especially when little training data is present, and showed that using TASK2VEC to select an expert from a collection can sensibly improve test performance while adding only a small overhead to the training process.

Meta-learning on the space of tasks is an important step toward general artificial intelligence. In this work, we introduce a way of dealing with thousands of tasks, enough to enable reconstruct a topology on the task space, and to test meta-learning solutions. The current experiments highlight the usefulness of our methods. Even so, our collection does not capture the full complexity and variety of tasks that one may encounter in real-world situations. Future work should further test effectiveness, robustness, and limitations of the embedding on larger and more diverse collections.

CHAPTER 6

Critical Learning Periods in Deep Networks

Critical periods are time windows of early post-natal development during which sensory deficits can lead to permanent skill impairment (Kandel et al., 2013). Researchers have documented critical periods affecting a range of species and systems, from visual acuity in kittens (Wiesel and Hubel, 1963b; Wiesel, 1982) to song learning in birds (Konishi, 1985). Uncorrected eye defects (*e.g.*, strabismus, cataracts) during the critical period for visual development lead to amblyopia in one in fifty adults.

The cause of critical periods is ascribed to the biochemical modulation of windows of neuronal plasticity (Hensch, 2004). However, in this chapter we show that deep neural networks (DNNs), while completely devoid of such regulations, respond to sensory deficits in ways similar to those observed in humans and animal models. This surprising result suggests that critical periods may arise from information processing, rather than biochemical, phenomena, and that tasks close to those that we already solved can still be unreachable during training due to emergent properties of DNNs.

In this chapter, we study critical period phenomena in DNNs using the Information in the Weights. We show that, counterintuitively, the information in the weights does not increase monotonically during training. Instead, a rapid growth in information ("memorization phase") is followed by a *reduction of information* ("reorganization" or "forgetting" phase), even as classification performance keeps increasing. This behavior is consistent across different tasks and network architectures. Critical periods are centered in the memorization phase.

Our findings indicate that the early transient is critical in determining the final solution of the optimization associated with training an artificial neural network. In particular, the

effects of sensory deficits during a critical period cannot be overcome, no matter how much additional training is performed. Yet most theoretical studies have focused on the network behavior after convergence (Representation Learning) or on the asymptotic properties of the optimization scheme used for training (SGD).

To study this early phase, using Fisher Information, we enstablish a connection between Information in the Weights and the *effective connectivity* of a network during training, and introduce the notion of *Information Plasticity* in learning. Information Plasticity is maximal during the memorization phase, and decreases in the reorganization phase. We show that deficit sensitivity during critical periods correlates strongly with the effective connectivity.

When considered in conjunction with the results in the previous chapters, our findings indicate that forgetting (reducing information in the weights) is critical to achieving invariance to nuisance variability as well as independence of the components of the representation, but comes at the price of reduced adaptability later in the training. We also hypothesize that the loss of physical connectivity in biology (neural plasticity) could be a consequence, rather than a cause, of the loss of Information Plasticity, which depends on how the information is distributed throughout a network during the early stages of learning. These results also shed light on the common practice of pre-training a model on a task and then fine-tune it for another, one of the most rudimentary forms of transfer learning. Our experiments show that, rather than helpful, pre-training can be detrimental, even if the tasks are similar (*e.g.*, same labels, slightly blurred images).

6.1 Deep Neural Networks have Critical Learning Periods

A notable example of critical period-related deficit, commonly affecting humans, is amblyopia (reduced visual acuity in one eye) caused by cataracts during infancy or childhood (Taylor et al., 1979; von Noorden, 1981). Even after surgical correction of cataracts, the ability of the patients to regain normal acuity in the affected eye depends both on the duration of the deficit and on its age of onset, with earlier and longer deficits causing more severe effects. In this section, we aim to study the effects of similar deficits in DNNs. To do so, we train

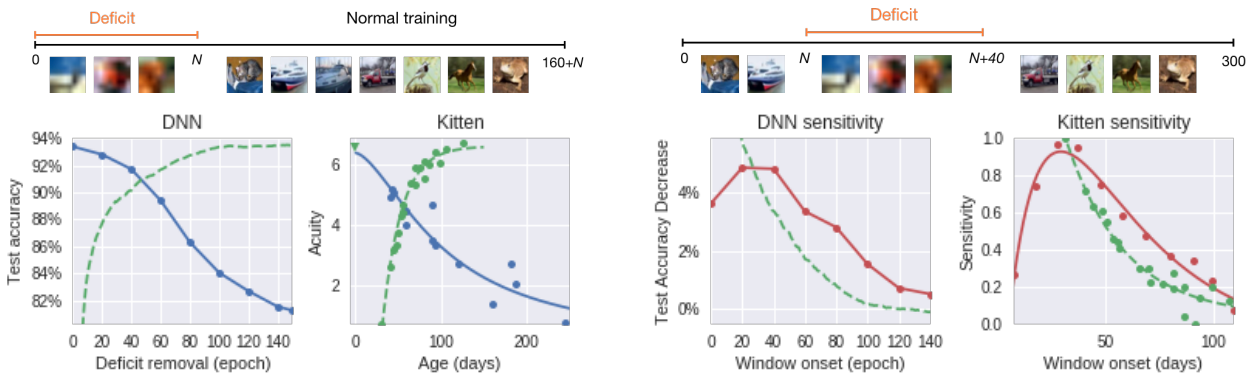


Figure 6.1: **DNNs exhibit critical periods. (A)** Final accuracy achieved by a CNN trained with a cataract-like deficit as a function of the training epoch N at which the deficit is removed (solid line). Performance is permanently impaired if the deficit is not corrected early enough, regardless of how much additional training is performed. As in animal models, critical periods coincide with the early learning phase during which, in the absence of deficits, test accuracy would rapidly increase (dashed). **(B)** For comparison, we report acuity for kittens monocularly deprived since birth and tested at the time of eye-opening (solid), and normal visual acuity development (in kittens) as a function of their age (dashed) (Giffin and Mitchell, 1978; Mitchell, 1988). **Sensitivity during learning: (C)** Final test accuracy of a DNN as a function of the onset of a short 40-epoch deficit. The decrease in the final performance can be used to measure the sensitivity to deficits. The most sensitive epochs corresponds to the early rapid learning phase, before the test error (dashed line) begins to plateau. Afterwards, the network is largely unaffected by the temporary deficit. **(D)** This can be compared with changes in the degree of functional disconnection (normalized numbers of V1 monocular cells disconnected from the contralateral eye) as a function of the kittens' age at the onset of a 10-12-day deficit window (Olson and Freeman, 1980). Dashed lines are as in A and B respectively, up to a re-scaling of the y-axis.

a standard All-CNN architecture based on Springenberg et al. (2014) (see **??**) to classify objects in small 32×32 images from the CIFAR-10 dataset (Krizhevsky and Hinton, 2009). We train with SGD using an exponential annealing schedule for the learning rate. To simulate the effect of cataracts, for the first t_0 epochs the images in the dataset are downsampled to 8×8 and then upsampled back to 32×32 using bilinear interpolation, in practice blurring the image and destroying small-scale details.[1] After that, the training continues for 160 more epochs, giving the network time to converge and ensuring it is exposed to the same number of uncorrupted images as in the control ($t_0 = 0$) experiment.

DNNs exhibit critical periods: In Figure 6.1, we plot the final performance of a network affected by the deficit as a function of the epoch t_0 at which the deficit is corrected. We can readily observe the existence of a critical period: If the blur is not removed within the first 40-60 epochs, the final performance is severely decreased when compared to the baseline (up to a threefold increase in error). The decrease in performance follows trends commonly observed in animals, and may be qualitatively compared, for example, to the loss of visual acuity observed in kittens monocularly deprived from birth as a function of the length of the deficit (Mitchell, 1988).

We can measure more accurately the sensitivity to a blur deficit during learning by introducing the deficit in a short window of constant length (40 epochs), starting at different epochs, and then measure the decrease in the DNN's final performance compared to the baseline (Figure 6.1). Doing this, we observe that the sensitivity to the deficit peaks in the central part of the early rapid learning phase (at around 30 epochs), while introducing the deficit later produces little or no effect. A similar experiment performed on kittens, using a window of 10-12 days during which the animals are monocularly deprived, again shows a remarkable similarity between the profiles of the sensitivity curves (Olson and Freeman, 1980).

High-level deficits are not associated with a critical period: A natural question is whether any change in the input data distribution will have a corresponding critical period

[1] We employed this method, instead of a simpler Gaussian blur, since it has a very similar effect and makes the quantification of information loss clearer.

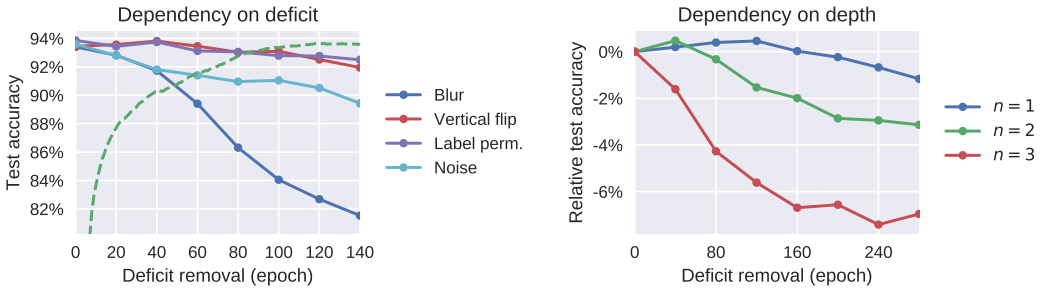


Figure 6.2: **(Left) High-level perturbations do not induce a critical period.** When the deficit only affects high-level features (vertical flip of the image) or the last layer of the CNN (label permutation), the network does not exhibit critical periods (test accuracy remains largely flat). On the other hand, a sensory deprivation-like deficit (image is replaced by random noise) does cause a deficit, but the effect is less severe than in the case of image blur. **(Right) Dependence of the critical period profile on the network's depth.** Adding more convolutional layers increases the effect of the deficit during its critical period (shown here is the decrease in test accuracy due to the deficit with respect to the test accuracy reached without deficits).

for learning. This is not the case for neuronal networks, which remain plastic enough to adapt to high-level changes in sensory processing (Daw, 2014). For example, it is well-reported that even adult humans can rapidly adapt to certain drastic changes, such as the inversion of the visual field (Stratton, 1896; Kohler, 1964). In Figure 6.2, we observe that DNNs are also largely unaffected by high-level deficits – such as vertical flipping of the image, or random permutation of the output labels: After deficit correction, the network quickly recovers its baseline performance. This hints at a finer interplay between the structure of the data distribution and the optimization algorithm, resulting in the existence of a critical period.

Sensory deprivation: We now apply to the network a more drastic deficit, where each image is replaced by white noise. Figure 6.2 shows hows this extreme deficit exhibits a remarkably less severe effect than the one obtained by only blurring images: Training the network with white noise does not provide any information on the natural images, and results in milder effects than those caused by a deficit (*e.g.*, image blur), which instead conveys *some* information, but leads the network to (incorrectly) learn that no fine structure is present in the images. A similar effect has been observed in animals, where a period of early sensory deprivation (dark-rearing) can lengthen the critical period and thus cause less severe effects than those documented in light-reared animals (Mower, 1991).

Architecture, depth, and learning rate annealing: Figure 6.3 shows that a fully-connected network trained on the MNIST digit classification dataset also shows a critical period for the image blur deficit. Therefore, the convolutional structure is not necessary, nor is the use of natural images. Similarly, a ResNet-18 trained on CIFAR-10 also has a critical period, which is also remarkably sharper than the one found in a standard convolutional network (Figure 6.1). This is especially interesting, since ResNets allow for easier backpropagation of gradients to the lower layers, thus suggesting that the critical period is not caused by vanishing gradients. However, Figure 6.2 (Right) shows that the presence of a critical period does indeed depend critically on the depth of the network. In Figure 6.3, we confirm that a critical period exists even when the network is trained with a constant learning rate, and therefore cannot be explained by an annealed learning rate in later epochs.

Optimization method and weight decay: Figure 6.3 (Bottom Right) shows that when using Adam as the optimization scheme, which renormalizes the gradients using a running mean of their first two moments, we still observe a critical period similar to that of standard SGD. However, changing the hyperparameters of the optimization can change the shape of the critical period: In Figure 6.3 (Bottom Left) we show that increasing weight decay makes critical periods longer and less sharp. This can be explained as it both slows the convergence of the network, and it limits the ability of higher layers to change to overcome the deficit, thus encouraging lower layers to also learn new features.

6.2 The role of Information in Critical Periods

We have established empirically that, in animals and DNNs alike, the initial phases of training are critical to the outcome of the training process. In animals, this strongly relates to changes in the brain architecture of the areas associated with the deficit (Daw, 2014). This is inevitably different in artificial networks, since their connectivity is formally fixed at all times during training. However, not all the connections are equally useful to the network: Consider a network encoding the approximate posterior distribution $p_w(y|x)$, parameterized by the weights w, of the task variable y given an input image x. The dependency of the final output from a specific connection can be estimated by perturbing the corresponding weight and looking at the magnitude of the change in the final distribution. Specifically, given a perturbation $w' = w + \delta w$ of the weights, the discrepancy between the $p_w(y|x)$ and the perturbed network output $p_{w'}(y|x)$ can be measured by their Kullback-Leibler divergence, which, to second-order approximation, is given by:

$$\mathbb{E}_x \, \mathrm{KL}(p_{w'}(y|x) \parallel p_w(y|x)) = \delta w \cdot F \delta w + o(\delta w^2),$$

where the expectation over x is computed using the empirical data distribution $\hat{Q}(x)$ given by the dataset, and

$$F := \mathbb{E}_{x \sim \hat{Q}(x)} \mathbb{E}_{y \sim p_w(y|x)} [\nabla_w \log p_w(y|x) \nabla_w \log p_w(y|x)^T]$$

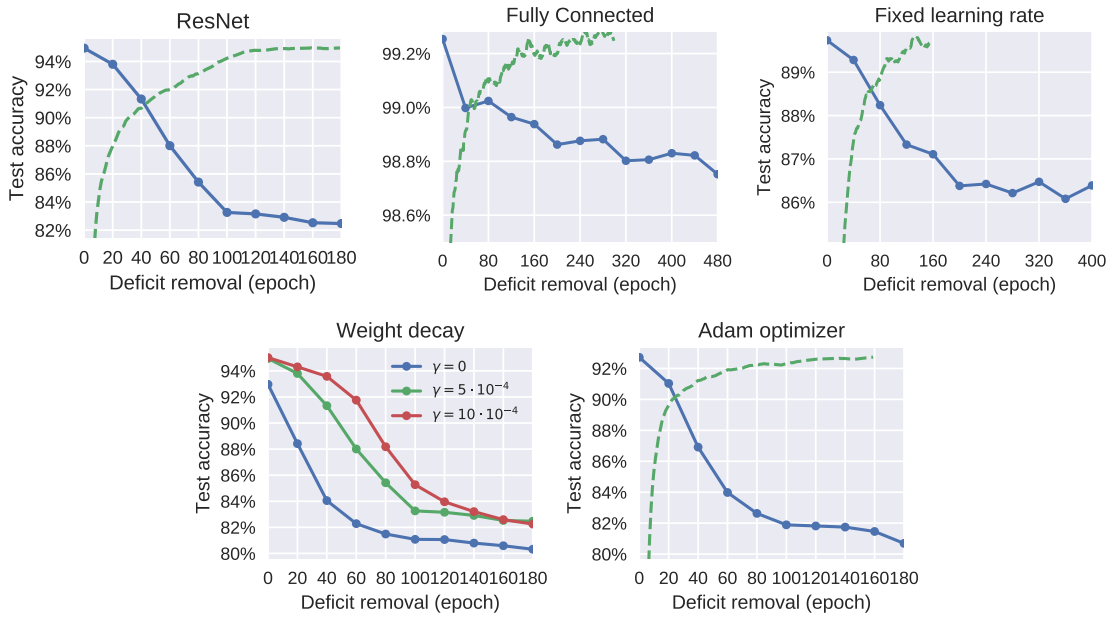


Figure 6.3: **Critical periods in different DNN architectures and optimization schemes. (Left)** Effect of an image blur deficit in a ResNet architecture trained on CIFAR-10 with learning rate annealing and **(Center)** in a deep fully-connected network trained on MNIST with a fixed learning rate. Different architectures, using different optimization methods and trained on different datasets, still exhibit qualitatively similar critical period behavior. **(Right)** Same experiment as in Figure 6.1, but using a fixed learning rate instead of an annealing scheme. Although the time scale of the critical period is longer, the trends are similar, supporting the notion that critical periods cannot be explained solely in terms of the loss landscape of the optimization. **(Bottom Left)** Networks trained without weight decay have shorter and sharper critical periods. Gradually increasing the weight decay makes the critical period longer, until the point where it stops training properly. **(Bottom Right)** When using a different optimization method (Adam) we obtain similar results as with standard SGD.

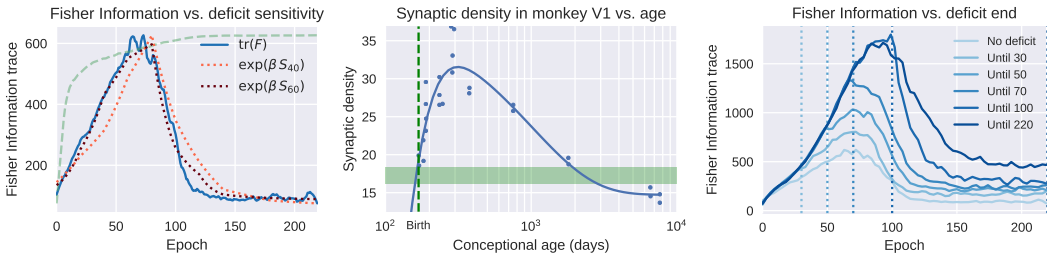


Figure 6.4: **Critical periods in DNNs are traced back to changes in the Fisher Information. (Left)** Trace of the Fisher Information of the network weights as a function of the training epoch (blue line), showing two distinct phases of training: First, information sharply increases, but once test performance starts to plateau (green line), the information in the weights decreases during a "consolidation" phase. Eventually less information is stored, yet test accuracy improves slightly (green line). The weights' Fisher Information correlates strongly with the network's sensitivity to critical periods, computed as in Figure 6.1 using both a window size of 40 and 60, and fitted here to the Fisher Information using a simple exponential fit. **(Center)** Recalling the connection between FIM ad connectivity, we may compare it to synaptic density during development in the visual cortex of macaques (Rakic et al., 1986). Here too, a rapid increase in connectivity is followed by elimination of synapses (pruning) continuing throughout life. **(Right)** Effects of a critical period-inducing blurring deficit on the Fisher Information: The impaired network uses more information to solve the task, compared to training in the absence of a deficit, since it is forced to memorize the labels case by case.

is the Fisher Information Matrix (FIM). The FIM can thus be considered a local metric measuring how much the perturbation of a single weight (or a combination of weights) affects the output of the network (Amari and Nagaoka, 2000b). In particular, weights with low Fisher Information can be changed or "pruned" with little effect on the network's performance. This suggests that the Fisher Information can be used as a measure of the effective connectivity of a DNN, or, more generally, of the "synaptic strength" of a connection (Kirkpatrick et al., 2017b). Finally, the FIM is also a semi-definite approximation of the Hessian of the loss function (**?**) and hence of the curvature of the loss landscape at a particular point w during training, providing an elegant connection between the FIM and the optimization procedure (Amari and Nagaoka, 2000b), which we will also employ later.

Unfortunately, the full FIM is too large to compute. Rather, we use its trace to measure the global or layer-wise connection strength, which we can compute efficiently using (**??**):

$$\mathrm{tr}(F) = \mathbb{E}_{x\sim\hat{Q}(x)}\mathbb{E}_{y\sim p_w(y|x)}[\|\nabla_w \log p_w(y|x)\|^2].$$

In order to capture the behavior of the off-diagonal terms, we also tried computing the log-determinant of the full matrix using the Kronecker-Factorized approximation of Martens and Grosse (2015), but we observed the same qualitative trend as the trace. Since the FIM is a local measure, it is very sensitive to the irregularities of the loss landscape. Therefore, in this section we mainly use ResNets, which have a relatively smooth landscape (Li et al., 2017). For other architectures we use instead a more robust estimator of the FIM based on the injection of noise in the weights (Achille and Soatto, 2018b), also described in **??**.

Two phases of learning: As its name suggests, the FIM can be thought as a measure of the quantity of information about the training data that is contained in the model (Fisher, 1925). Based on this, one would expect the overall strength of the connections to increase monotonically as we acquire information from experience. However, this is not the case: While during an initial phase the network acquires information about the data, which results in a large increase in the strength of the connections, once the performance in the task begins to plateau, the network starts decreasing the overall strength of its connections. However, this does not correspond to a reduction in performance, rather, performance keeps

slowly improving. This can be seen as a "forgetting", or "compression" phase, during which redundant connections are eliminated and non-relevant variability in the data is discarded. It is well-established how the elimination ("pruning") of unnecessary synapses is a fundamental process during learning and brain development (Rakic et al., 1986) (Figure 6.4, Center); in Figure 6.4 (Left) an analogous phenomenon is clearly and quantitatively shown for DNNs.

Strikingly, these changes in the connection strength are closely related to the sensitivity to critical-period-inducing deficits such as image blur, computed using the "sliding window" method as in Figure 6.1. In Figure 6.4 we see that the sensitivity closely follows the trend of the FIM. This is remarkable since the FIM is a local quantity computed at a single point during the training of a network in the absence of deficit, while sensitivity during a critical period is computed, using test data, at the end of the impaired network training. Figure 6.4 (Right) further emphasizes the effect of deficits on the FIM: in the presence of a deficit, the FIM grows and remains substantially higher even after the deficit is removed. This may be attributed to the fact that, when the data are so corrupted that classification is impossible, the network is forced to memorize the labels, therefore increasing the quantity of information needed to perform the same task.

Layer-wise effects of deficits: A layer-wise analysis of the FIM sheds further light on how the deficit affects the network. When the network (in this case All-CNN, which has a clearer division among layers than ResNet) is trained without deficits, the most important connections are in the intermediate layers (Figure 6.5, Left), which can process the input CIFAR-10 image at the most informative intermediate scale. However, if the network is initially trained on blurred data (Figure 6.5, top right), the strength of the connections is dominated by the top layer (Layer 6). This is to be expected, since the low-level and mid-level structures of the images are destroyed, making the lower layers ineffective. However, if the deficit is removed early in the training (Figure 6.5, top center), the network manages to "reorganize", reducing the information contained in the last layer, and, at the same time, increasing the information in the intermediate layers. We refer to these phenomena as changes in "Information Plasticity". If, however, the data change occurs after the consolidation phase, the network is unable to change its effective connectivity: The connection strength of each

layer remains substantially constant. The network has lost its Information Plasticity and is past its critical period.

Critical periods as bottleneck crossings: The analysis of the FIM also sheds light on the geometry of the loss function and the learning dynamics. Since the FIM can be interpreted as the local curvature of the residual landscape, Fig. 4 shows that learning entails crossing "bottlenecks:" In the initial phase the network enters regions of high curvature (high Fisher Information), and once consolidation begins, the curvature decreases, allowing it to cross the bottleneck and enter the valley below. If the statistics change after crossing the bottleneck, the network is trapped. In this interpretation, the early phases of convergence are critical in leading the network towards the "right" final valley. The end of critical periods comes after the network has crossed all bottlenecks (and thus learned the features) and entered a wide valley (region of the weight space with low curvature, or low Fisher Information).

6.3 Discussion and Related Work

Critical periods have thus far been considered an exclusively biological phenomenon. At the same time, the analysis of DNNs has focused on asymptotic properties and neglected the initial transient behavior. To the best of our knowledge, we are the first to show that artificial neural networks exhibit critical period phenomena, and to highlight the critical role of the transient in determining the asymptotic performance of the network. Inspired by the role of synaptic connectivity in modulating critical periods, we introduce the use of Fisher Information to study this initial phase. We show that the initial sensitivity to deficits closely follows changes in the FIM, both global, as the network first rapidly increases and then decreases the amount of stored information, and layer-wise, as the network "reorganizes" its effective connectivity in order to optimally process information.

Our work naturally relates to the extensive literature on critical periods in biology. Despite artificial networks being an extremely reductionist approximation of neuronal networks, they exhibit behaviors that are qualitatively similar to the critical periods observed in hu-

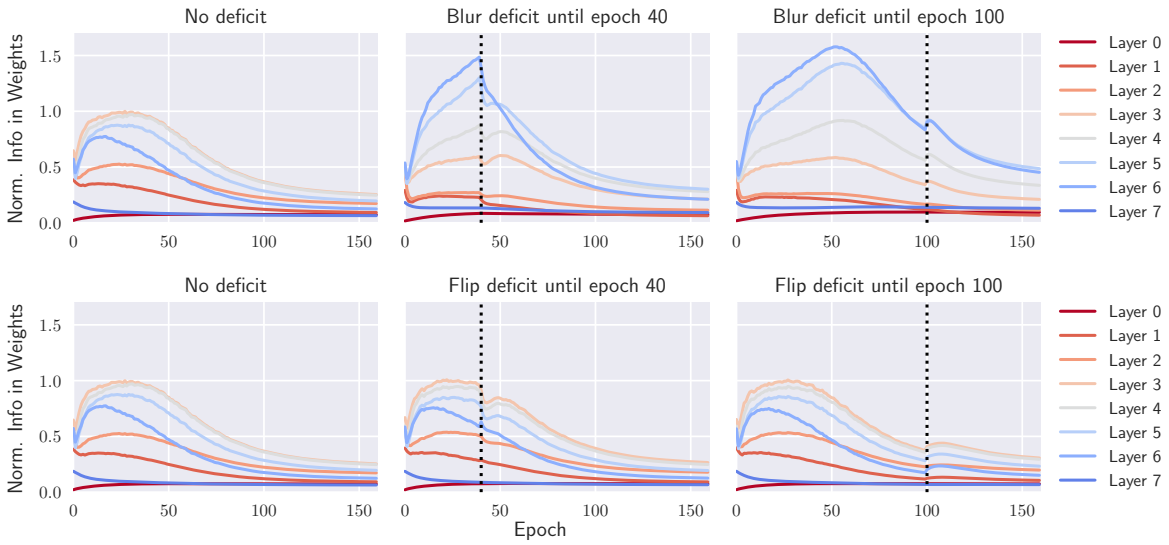


Figure 6.5: Normalized quantity of information contained in the weights of each layer as a function of the training epoch. **(Top Left)** In the absence of deficits, the network relies mostly on the middle layers (3-4-5) to solve the task. **(Top Right)** In the presence of an image blur deficit until epoch 100, more resources are allocated to the higher layers (6-7) rather than to the middle layers. The blur deficit destroys low- and mid-level features processed by those layers, leaving only the global features of the image, which are processed by the higher layers. Even if the deficit is removed, the middle layers remain underdeveloped. **(Top Center)** When the deficit is removed at an earlier epoch, the layers can partially reconfigure (notice, *e.g.*, the fast loss of information of layer 6), resulting in less severe long-term consequences. We refer to the redistribution of information and the relative changes in effective connectivity as "Information Plasticity". **(Bottom row)** Same plots, but using a vertical flip deficit, which does not induce a critical period. As expected, the quantity of information in the layers is not affected.

man and animal models. Our information analysis shows that the initial rapid memorization phase is followed by a loss of Information Plasticity which, counterintuitively, further improves the performance. On the other hand, when combined with the analysis of Chapter 4, this suggests that a "forgetting" phase may be desirable, or even necessary, in order to learn robust, nuisance-invariant representations.

The existence of two distinct phases of training has been observed and discussed by Shwartz-Ziv and Tishby (2017), although their analysis builds on the (Shannon) information of the *activations*, rather than the (Fisher) information in the *weights*. On a multi-layer perceptron (MLP), Shwartz-Ziv and Tishby (2017) empirically link the two phases to a sudden increase in the gradients' covariance. It may be tempting to compare these results with our Fisher Information analysis. However, it must be noted that the FIM is computed using the gradients with respect to the model prediction, not to the ground truth label, leading to important qualitative differences. In **??**, we show that the covariance and norm of the gradients exhibit no clear trends during training with and without deficits, and, therefore, unlike the FIM, do not correlate with the sensitivity to critical periods. Nonetheless, a connection between our FIM analysis and the information in the activations can be established based on Section 4.5, which shows that the FIM of the weights can be used to bound the information in the activations. In fact, we may intuitively expect that "pruning" of connections naturally leads to loss of information in the corresponding activations. Thus, our analysis corroborates and expands on some of the claims of Shwartz-Ziv and Tishby (2017), while using an independent framework.

Aside from being more closely related to the deficit sensitivity during critical periods, Fisher Information also has a number of technical advantages: Its diagonal is simple to estimate, even on modern state-of-the-art architectures and compelling datasets, and it is less sensitive to the choice estimator of mutual information, avoiding some of the most common criticism of the use of information quantities in the analysis of deep learning models. Finally, the FIM allows us to probe fine changes in the effective connectivity across the layers of the network (Figure 6.5), which are not visible in Shwartz-Ziv and Tishby (2017).

A complete analysis of the activations should account not only for the amount of in-

formation (both task- and nuisance-related), but also for its accessibility, *e.g.*, how easily task-related information can be extracted by a linear classifier. Following a similar idea, Montavon et al. (2011) studied the layer-wise, or "spatial" (but not temporal) evolution of the simplicity of the representation by performing a principal component analysis (PCA) of a radial basis function (RBF) kernel embedding of each layer representation. They showed that, on a multi-layer perceptron, task-relevant information increasingly concentrate on the first principal components of the representation's embedding, implying that they become more easily "accessible" layer after layer, while nuisance information (when codified at all) is encoded in the remaining components. In our work, instead, we focus on the temporal evolution of the weights. However, it is important to notice how a network with simpler weights (as measured by the FIM) also requires a simpler smooth representation (as measured, *e.g.*, by the RBF embedding) in order to operate properly, since it needs to be resistant to perturbations of the weights. Thus our analysis is wholly compatible with the intuitions of Montavon et al. (2011). It would be interesting to study the joint spatio-temporal evolution of the network using both frameworks at once.

One advantage of focusing on the information of the weights rather than on the activations, or the behavior of the network, is to have a readout of the "effective connectivity" during critical periods, which can be compared to similar readouts in animals. In fact, "behavioral" readouts upon deficit removal, both in artificial and neuronal networks, can potentially be confounded by deficit-coping changes at different levels of the visual pathways (Daw, 2014; Knudsen, 2004). On the other hand, deficits in deprived animals correlate strongly with the abnormalities in the circuitry of the visual pathways, which we characterize in DNNs using the FIM to study its "effective connectivity", *i.e.*, the connections that are actually employed by the network to solve the task. Sensitivity to critical periods and the trace of the Fisher Information peak at the same epochs, in accord with the evidence that skill development and critical periods in neuronal networks are modulated by changes (generally experience-dependent) in synaptic plasticity (Knudsen, 2004; Hensch, 2004). Our layer-wise analysis of the Fisher Information (Figure 6.5) also shows that visual deficits reinforce higher layers to the detriment of intermediate layers, leaving low-level layers virtually untouched. If

the deficit is removed after the critical period ends, the network is not able to reverse these effects. Although the two systems are radically different, a similar response can be found in the visual pathways of animal models: Lower levels (*e.g.*, retina, lateral geniculate nucleus) and higher-level visual areas (*e.g.*, V2 and post-V2) show little remodeling upon deprivation, while most changes happen in different layers of V1 (Wiesel and Hubel, 1963a; Hendrickson et al., 1987).

An insightful interpretation of critical periods in animal models was proposed by Knudsen (2004): The initial connections of neuronal networks are unstable and easily modified (highly plastic), but as more "samples" are observed, they change and reach a more stable configuration which is difficult to modify. Learning can, however, still happen within the newly created connectivity pattern. This is largely compatible with our findings: Sensitivity to critical-period-inducing deficits peaks when connections are remodeled (Figure 6.4, Left), and different connectivity profiles are observed in networks trained with and without a deficit (Figure 6.5). Moreover, high-level deficits such as image-flipping and label permutation, which do not require radical restructuring of the network's connections in order to be corrected, do not exhibit a critical period.

Applying a deficit at the beginning of the training may be compared to the common practice of pre-training, which is generally found to improve the performance of the network. Erhan et al. (2010) study the somewhat related, but now seldom used, practice of layer-wise unsupervised pre-training, and suggest that it may act as a regularizer by moving the weights of the network towards an area of the loss landscape closer to the attractors for good solutions, and also that early examples have a stronger effect in steering the network towards particular solutions. Here, we have shown that pre-training on blurred data can have the opposite effect; *i.e.*, it can severely decrease the final performance of the network. However, in our case, interpreting the deficit's effect as moving the network close to a bad attractor is difficult to reconcile with the smooth transition observed in the critical periods since the network would either converge to this attractor, and thus have low accuracy, or escape completely.

Instead, we can reconcile our experiments with the geometry of the loss function by intro-

ducing a different explanation based on the interpretation of the FIM as an approximation of the local curvature. Figure 6.4 suggests that SGD encounters two different phases during the network training: At first, the network moves towards high-curvature regions of the loss landscape, while in the second phase the curvature decreases and the network eventually converges to a flat minimum (as observed in Keskar et al. (2017)). We can interpret these as the network crossing narrow bottlenecks during its training in order to learn useful features, before eventually entering a flat region of the loss surface once learning is completed and ending up trapped there. When combining this assumption with our deficit sensitivity analysis, we can hypothesize that the critical period occurs precisely upon crossing of this bottleneck. It is also worth noticing how there is evidence that convergence to flat minima (minima with low curvature) in a DNN correlates with a good generalization performance (Hochreiter and Schmidhuber, 1997; Li et al., 2017; Chaudhari et al., 2017; Keskar et al., 2017). Indeed, using this interpretation, Figure 6.4 (Right) tells us that networks more affected by the deficit converge to sharper minima. However, we have also found that the performance of the network is already mostly determined during the early "sensitive" phase. The final sharpness at convergence may thus be an epiphenomenon rather than the cause of good generalization.

Bibliography

iMaterialist Challenge (Fashion) at FGVC5 workshop, CVPR. https://www.kaggle.com/c/imaterialist-challenge-fashion-2018. URL https://www.kaggle.com/c/imaterialist-challenge-fashion-2018.

Salisu Mamman Abdulrahman, Pavel Brazdil, Jan N van Rijn, and Joaquin Vanschoren. Speeding up algorithm selection using average ranking and active testing by introducing runtime. *Machine learning*, 107(1):79–108, 2018.

Alessandro Achille and Alessandro Berarducci. A vietoris-smale mapping theorem for the homotopy of hyperdefinable sets. *Selecta Mathematica*, pages 1–29, 4 2018. ISSN 1022-1824. doi: 10.1007/s00029-018-0413-3.

Alessandro Achille and Stefano Soatto. Emergence of Invariance and Disentangling in Deep Representations. *Proceedings of the ICML Workshop on Principled Approaches to Deep Learning*, 2017.

Alessandro Achille and Stefano Soatto. A separation principle for control in the age of deep learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1), 2018a. doi: 10.1146/annurev-control-060117-105140.

Alessandro Achille and Stefano Soatto. Emergence of invariance and disentanglement in deep representations. *Journal of Machine Learning Research*, 19(1):1947–1980, 2018b.

Alessandro Achille and Stefano Soatto. Information dropout: Learning optimal representations through noisy computation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018. doi: 10.1109/TPAMI.2017.2784440.

Alessandro Achille, Tom Eccles, Loic Matthey, Chris Burgess, Nicholas Watters, Alexander Lerchner, and Irina Higgins. Life-long disentangled representation learning with cross-domain latent homologies. In *Advances in Neural Information Processing Systems 31*, pages 9895–9905. 2018.

Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. Task2Vec: Task Embedding for Meta-Learning. *arXiv:1902.03545*, 2019a.

Alessandro Achille, Giovanni Paolini, Glen Mbeng, and Stefano Soatto. The Information Complexity of Learning Tasks, their Structure and their Distance. *arXiv:1904.03292*, 2019b.

Alessandro Achille, Matteo Rovere, and Stefano Soatto. Critical learning periods in deep neural networks. *Proc. of the International Conference on Learning Representations (ICLR)*, 2019.

Kofi P Adragni and R Dennis Cook. Sufficient dimension reduction and prediction in regression. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 367(1906):4385–4405, 2009.

Alexander A Alemi, Ian Fischer, Joshua V Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv preprint arXiv:1612.00410*, 2016.

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2): 251–276, 1998.

Shun-Ichi Amari and Hiroshi Nagaoka. Methods of information geometry, volume 191 of translations of mathematical monographs. *American Mathematical Society*, 13, 2000a.

Shun-ichi Amari and Hiroshi Nagaoka. *Methods of information geometry*, volume 191 of *Translations of Mathematical Monographs*. American Mathematical Society and Oxford University Press, 2000b.

Bernard Ans and Stéphane Rousset. Avoiding catastrophic forgetting by coupling two reverberating neural networks. *Comptes Rendus de l'Académie des Sciences - Series III - Sciences de la Vie*, 320(12):989–997, 1997.

Fabio Anselmi, Lorenzo Rosasco, and Tomaso Poggio. On invariance and selectivity in representation learning. *Information and Inference*, 5(2):134–158, 2016.

Sanjeev Arora, Rong Ge, Behnam Neyshabur, and Yi Zhang. Stronger generalization bounds for deep nets via a compression approach. *arXiv preprint arXiv:1802.05296*, 2018.

Allison Auchter, Lawrence K Cormack, Yael Niv, Francisco Gonzalez-Lima, and Marie H Monfils. Reconsolidation-extinction interactions in fear memory attenuation: the role of inter-trial interval variability. *Frontiers in behavioral neuroscience*, 11:2, 2017.

Charles Beattie, Joel Z. Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, Julian Schrittwieser, Keith Anderson, Sarah York, Max Cant, Adam Cain, Adrian Bolton, Stephen Gaffney, Helen King, Demis Hassabis, Shane Legg, and Stig Petersen. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

Anthony J Bell and Terrence J Sejnow. An information-maximization approach to blind separation and blind deconvolution. *Neural Computation*, 7:1129–1159, 1995.

Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35 (8):1798–1828, 2013.

Sterling K. Berberian. Borel spaces, April 1988.

Joan Bruna and Stéphane Mallat. Classification with scattering operators. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1561–1566, 2011.

Christopher P. Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-VAE. *NIPS Workshop of Learning Disentangled Features*, 2017.

Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. *Proc. of the International Conference on Learning Representations (ICLR)*, 2018.

Pratik Chaudhari, Anna Choromanska, Stefano Soatto, Yann LeCun, Carlo Baldassi, Christian Borgs, Jennifer Chayes, Levent Sagun, and Riccardo Zecchina. Entropy-sgd: Biasing

gradient descent into wide valleys. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Joseph Cichon and Wen-Biao Gan. Branch-specific dendritic ca2+ spikes cause persistent synaptic plasticity. *Nature*, 520(7546):180–185, 2015.

Pierre Comon. Independent component analysis, a new concept? *Signal Processing*, 36: 287–314, 1994.

Nigel W Daw. *Visual Development*. Springer, New York, NY, 3rd edition, 2014.

Gintare Karolina Dziugaite and Daniel M Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. *arXiv preprint arXiv:1703.11008*, 2017.

Harrison Edwards and Amos Storkey. Towards a neural statistician. *arXiv preprint arXiv:1606.02185*, 2016.

Dumitru Erhan, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11(Feb):625–660, 2010.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymir Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. *arxiv*, 2018.

Nancy L. Etcoff and John J. Magee. Categorical perception of facial expressions. *Cognition*, 44:227–240, 1992.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *arXiv preprint arXiv:1703.03400*, 2017.

Ronald Aylmer Fisher. Theory of statistical estimation. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 22, pages 700–725. Cambridge University Press, 1925.

Maximilian Poggio Tomaso Miller Earl K. Freedman, David J. Riesenhuber. Categorical representation of visual stimuli in the primate prefrontal cortex. *Science*, 291:312–316, 2001.

Tommaso Furlanello, Jiaping Zhao, Andrew M Saxe, Laurent Itti, and Bosco S Tjan. Active long term memory networks. *arXiv preprint arXiv:1606.02355*, 2016.

Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015.

Marta Garnelo, Kai Arulkumaran, and Murray Shanahan. Towards deep symbolic reinforcement learning. *arXiv preprint arXiv:1609.05518*, 2016.

Fred Giffin and Donald E Mitchell. The rate of recovery of vision after early monocular deprivation in kittens. *The Journal of Physiology*, 274(1):511–537, 1978.

Peter D Grünwald. *The minimum description length principle*. MIT press, 2007.

Balazs Gulyas, David Ottoson, and Per E. Roland. *Functional Organisation of the Human Visual Cortex*. Wenner–Gren International Series, 1993.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *ICCV*, 2015.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

Anita E Hendrickson, JA Movshon, Howard M Eggers, Martin S Gizzi, RG Boothe, and Lynne Kiorpes. Effects of early unilateral blur on the macaque's visual system. ii. anatomical observations. *Journal of Neuroscience*, 7(5):1327–1339, 1987.

Takao K Hensch. Critical period regulation. *Annuual review of neuroscience*, 27:549–579, 2004.

Tom Heskes. On "natural" learning and pruning in multilayered perceptrons. *Neural Computation*, 12(4):881–901, 2000.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. β-VAE: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2017a.

Irina Higgins, Loic Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017b.

Irina Higgins, Arka Pal, Andrei Rusu, Loic Matthey, Christopher Burgess, Alexander Pritzel, Matthew Botvinick, Charles Blundell, and Alexander Lerchner. DARLA: Improving zero-shot transfer in reinforcement learning. *ICML*, 2017c.

Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the 6th annual conference on Computational learning theory*, pages 5–13. ACM, 1993.

Sepp Hochreiter and Jürgen Schmidhuber. Flat minima. *Neural Computation*, 9(1):1–42, 1997.

Alex D Holub, Max Welling, and Pietro Perona. Combining generative models and fisher kernels for object recognition. In *IEEE International Conference on Computer Vision*, volume 1, pages 136–143. IEEE, 2005.

Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Tommi S Jaakkola and David Haussler. Exploiting generative models in discriminative classifiers. In *Advances in Neural Information Processing Systems*, pages 487–493, 1999.

Tommi S Jaakkola, Mark Diekhans, and David Haussler. Using the Fisher kernel method to detect remote protein homologies. In *ISMB*, volume 99, pages 149–158, 1999.

Eric R Kandel, James H Schwartz, Thomas M Jessell, Steven A Siegelbaum, and A James Hudspeth. *Principles of Neural Science*. McGraw-Hill, New York, NY, 5th edition, 2013.

Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *Proceedings of the International Conference on Learning Representations*, 2017.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

Diederik P Kingma, Tim Salimans, and Max Welling. Variational dropout and the local reparameterization trick. In *Advances in Neural Information Processing Systems*, pages 2575–2583, 2015.

Diederik P Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*, pages 4743–4751, 2016.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *PNAS*, 114(13):3521–3526, 2017a.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, page 201611835, 2017b.

Eric I Knudsen. Sensitive periods in the development of the brain and behavior. *Journal of cognitive neuroscience*, 16(8):1412–1425, 2004.

Ivo Kohler. *The formation and transformation of the perceptual world.* Psychological Issues Monographs. International Universities Press, Inc., New York, NY, 1964.

Masakazu Konishi. Birdsong: from behavior to neuron. *Annual review of neuroscience*, 8 (1):125–170, 1985.

Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Technical report, University of Toronto*, 2009.

Brenden M. Lake, Tomer D. Ullman, Joshua B. Tenenbaum, and Samuel J. Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, pages 1–101, 2016.

Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Rui Leite, Pavel Brazdil, and Joaquin Vanschoren. Selecting classification algorithms with active testing. In *International workshop on machine learning and data mining in pattern recognition*, pages 117–131. Springer, 2012.

Hao Li, Zheng Xu, Gavin Taylor, and Tom Goldstein. Visualizing the loss landscape of neural nets. *arXiv preprint arXiv:1712.09913*, 2017.

Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes. Fisher-Rao metric, geometry, and complexity of neural networks. *arXiv preprint arXiv:1711.01530*, 2017.

Xiuwen Liu, Anuj Srivastava, and Kyle Gallivan. Optimal linear representations of images for object recognition. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 1, pages I–I. IEEE, 2003.

Yan Liu and Bharathi Jagadeesh. Neural selectivity in anterior inferotemporal cortex for morphed photographic images during behavioral classification or fixation. *J. Neurophysiol.*, 100:966–982, 2008.

Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1096–1104, 2016.

James Martens. New perspectives on the natural gradient method. *CoRR*, abs/1412.1193, 2014. URL `http://arxiv.org/abs/1412.1193`.

James Martens and Roger Grosse. Optimizing neural networks with kronecker-factored approximate curvature. *Proceedings of International Conference on Machine Learning*, 37:2408–2417, 2015.

Pyry Matikainen, Rahul Sukthankar, and Martial Hebert. Model recommendation for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2256–2263. IEEE, 2012.

David McAllester. A pac-bayesian tutorial with a dropout bound. *arXiv preprint arXiv:1307.2118*, 2013.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3): 419, 1995.

Kieran Milan, Joel Veness, James Kirkpatrick, Demis Hassabis, Anna Koop, and Michael Bowling. The forget-me-not process. *NIPS*, 2016.

Donald E Mitchell. The extent of visual recovery from early monocular or binocular visual deprivation in kittens. *The Journal of physiology*, 395(1):639–660, 1988.

Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pages 2204–2212, 2014.

Volodymyr Mnih, Koray Kavukcuoglu, David S Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig

Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

Grégoire Montavon, Mikio L Braun, and Klaus-Robert Müller. Kernel analysis of deep networks. *Journal of Machine Learning Research*, 12(Sep):2563–2581, 2011.

George D Mower. The effect of dark rearing on the time course of the critical period in cat visual cortex. *Developmental Brain Research*, 58(2):151–158, 1991.

Youssef Mroueh and Tom Sercu. Fisher GAN. In *Advances in Neural Information Processing Systems*, pages 2513–2523, 2017.

Louis A. Necker. Observations on some remarkable optical phaenomena seen in switzerland; and on an optical phaenomenon which occurs on viewing a figure of a crystal or geometrical solid. *London and Edinburgh Philosophical Magazine and Journal of Science*, 1(5):329–337, 1832.

Behnam Neyshabur, Ruslan R Salakhutdinov, and Nati Srebro. Path-sgd: Path-normalized optimization in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2422–2430, 2015.

Cuong V. Nguyen, Yingzhen Li, Thang D. Bui, and Richard E. Turner. Variational continual learning. *ICLR*, 2018.

Carl R Olson and Ralph D Freeman. Profile of the sensitive period for monocular deprivation in kittens. *Experimental Brain Research*, 39(1):17–21, 1980.

Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning. *ICLR*, 2015.

Florent Perronnin, Jorge Sánchez, and Thomas Mensink. Improving the fisher kernel for large-scale image classification. In *European Conference on Computer Vision*, pages 143–156. Springer, 2010.

Andrzej Przybyszewski. Vision: Does top-down processing help us to see? *Current Biology*, 8:135–139, 1998.

Pasko Rakic, Jean-Pierre Bourgeois, Maryellen F Eckenhoff, Nada Zecevic, and Patricia S Goldman-Rakic. Concurrent overproduction of synapses in diverse regions of the primate cerebral cortex. *Science*, 232(4747):232–235, 1986.

Jason Ramapuram, Magda Gregorova, and Alexandros Kalousis. Lifelong generative modeling. *arXiv preprint arXiv:1705.09847*, 2017.

Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2):285, 1990.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *ICML*, 32(2):1278–1286, 2014.

Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

Anthony Robins. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science*, 7(2):123–146, 1995.

Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. *arxiv*, 2016.

Paul Ruvolo and Eric Eaton. Ella: An efficient lifelong learning algorithm. *ICML*, 2013.

Jorge Sánchez, Florent Perronnin, Thomas Mensink, and Jakob Verbeek. Image classification with the fisher vector: Theory and practice. *International Journal of Computer Vision*, 105(3):222–245, 2013.

Craig Saunders, Alexei Vinokourov, and John S Shawe-taylor. String kernels, Fisher kernels and finite state automata. In *Advances in Neural Information Processing Systems*, pages 649–656, 2003.

Matthias Seeger. Learning with labeled and unlabeled data. Technical Report EPFL-REPORT-161327, Institute for Adaptive and Neural Computation, University of Edinburgh, 2000.

Ari Seff, Alex Beatson, Daniel Suo, and Han Liu. Continual learning in generative adversarial nets. *NIPS*, 2017.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. *NIPS*, 2017.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587): 484–489, 2016.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Michael R Smith, Logan Mitchell, Christophe Giraud-Carrier, and Tony Martinez. Recommending learning algorithms and their associated hyperparameters. *arXiv preprint arXiv:1407.1890*, 2014.

Stefano Soatto and Alessandro Chiuso. Visual representations: Defining properties and deep approximations. In *Proceedings of the International Conference on Learning Representations (ICLR)*. 2016.

Casper Kaae Sønderby, Jose Caballero, Lucas Theis, Wenzhe Shi, and Ferenc Huszár. Amortised map inference for image super-resolution. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

Greg Ver Steeg. Unsupervised learning via total correlation explanation. *arXiv preprint arXiv:1706.08984*, 2017.

George M Stratton. Some preliminary experiments on vision without inversion of the retinal image. *Psychological Review*, 3(6):611–617, 1896.

Milan Studenỳ and Jirina Vejnarová. The multiinformation function as a tool for measuring stochastic dependence. In *Learning in graphical models*, pages 261–297. Springer, 1998.

Ganesh Sundaramoorthi, Peter Petersen, V. S. Varadarajan, and Stefano Soatto. On the set of images modulo viewpoint and contrast changes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2009.

David Taylor et al. Critical period for deprivation amblyopia in children. *Transactions of the ophthalmological societies of the United Kingdom*, 99(3):432–439, 1979.

Naftali Tishby and Noga Zaslavsky. Deep learning and the information bottleneck principle. In *Information Theory Workshop (ITW), 2015 IEEE*, pages 1–5. IEEE, 2015.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. pages 368–377, 1999.

Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1521–1528. IEEE, 2011.

Laurens Van Der Maaten. Learning Discriminative Fisher Kernels. In *ICML*, volume 11, pages 217–224, 2011.

Grant Van Horn, Oisin Mac Aodha, Yang Song, Yin Cui, Chen Sun, Alex Shepard, Hartwig Adam, Pietro Perona, and Serge Belongie. The inaturalist species classification and detection dataset. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2018.

Nikolai K Vereshchagin and Paul MB Vitányi. Kolmogorov's structure functions and model selection. *IEEE Transactions on Information Theory*, 50(12):3265–3290, 2004.

Gunter K von Noorden. New clinical aspects of stimulus deprivation amblyopia. *American journal of ophthalmology*, 92(3):416–421, 1981.

Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology, 2011.

Sida Wang and Christopher Manning. Fast dropout training. In *International Conference on Machine Learning (ICML)*, pages 118–126, 2013.

Yu-Xiong Wang and Martial Hebert. Model recommendation: Generating object detectors from few samples. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1619–1628, 2015.

Satosi Watanabe. Information theoretical analysis of multivariate correlation. *IBM Journal of research and development*, 4(1):66–82, 1960.

Torsten N Wiesel. Postnatal development of the visual cortex and the influence of environment. *Nature*, 299(5884):583, 1982.

Torsten N Wiesel and David H Hubel. Single-cell responses in striate cortex of kittens deprived of vision in one eye. *Journal of neurophysiology*, 26(6):1003–1017, 1963a.

Torsten N Wiesel and David H Hubel. Effects of visual deprivation on morphology and physiology of cells in the cat's lateral geniculate body. *Journal of neurophysiology*, 26(6): 978–993, 1963b.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arxiv*, 2017.

Amir R Zamir, Alexander Sax, William Shen, Leonidas Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3712–3722, 2018.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. *ICML*, 2017.

Chiyuan Zhang, Samy Bengio, Moritz Hardt, Benjamin Recht, and Oriol Vinyals. Understanding deep learning requires rethinking generalization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Peng Zhang, Jiuling Wang, Ali Farhadi, Martial Hebert, and Devi Parikh. Predicting failures of vision systems. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3566–3573, 2014.

Xiaogang Wang Xiaoou Tang Ziwei Liu, Ping Luo. Deep learning face attributes in the wild. *International Conference on Computer Vision (ICCV)*, 2015.