AN INFORMATION GEOMETRIC PICTURE OF THE SPACE OF TASKS

Yansong Gao

A DISSERTATION

in

Applied Mathematics and Computational Science

Presented to the Faculties of the University of Pennsylvania

in

Partial Fulfillment of the Requirements for the

Degree of Doctor of Philosophy

2022

Supervisor of Dissertation

Pratik Chaudhari, Assistant Professor, Electrical and Systems Engineering

Graduate Group Chairperson

Robin Pemantle, Merriam Term Professor of Mathematics

Dissertation Committee

Vijay Balasubramanian, Cathy and Marc Lasry Professor, Physics and Astronomy
Pratik Chaudhari, Assistant Professor, Electrical and Systems Engineering
Kostas Daniilidis, Ruth Yalom Stone Professor, Computer and Information Science
Weijie Su, Associate Professor, Wharton Statistics and Data Science

ABSTRACT

AN INFORMATION GEOMETRIC PICTURE OF THE SPACE OF TASKS

Yansong Gao

Pratik Chaudhari

This dissertation seeks to address why deep learning models can be effectively applied to a wide range of tasks. Understanding the space of tasks lays the foundation for answering this question. We leverage information geometry—a well-established set of tools to gain a deeper understanding of the space of typical tasks and make the following contributions.

In Chapter 2, we formalize the free energy principle that addresses how to perform pre-training effectively. **The free energy principle identifies reconstruction as the canonical task that pre-training procedures should consider to improve the representation quality for multiple other tasks.** Leveraging insights from variational inference, the free energy principle foretold the effectiveness of reconstruction pre-training prior to its widespread adoption.

By leveraging optimal transportation, Chapter 3 establishes a sequence of interpolated tasks that evolves from pre-training to target tasks. The representation is then updated to align with the evolving data distribution. We refer to this process as *optimal coupled transfer*. The optimal coupled transfer enables the pre-trained model to traverse the shortest path in the space of tasks. **From an information geometric perspective, the length of this shortest path connecting two tasks gives rise to a unique definition of the distance between them.**

In the context of learning with unlabeled data $p(x)$ (Chapter 4), we can also harness the power of reconstruction. **A canonical approach to explore unlabeled data is to directly reconstruct the potential downstream tasks $p(x, y)$ in the space of tasks.** By leveraging the power of a *reference prior*, we reconstruct a pool of diverse tasks that encompass the typical downstream tasks without knowing the actual labels. Through empirical experiments, we demonstrate the effectiveness of our approach, achieving state-of-the-art results in self-supervised learning.

Statistical learning theory insights suggest that building a single model for all tasks (e.g., foundation models) may not be ideal. **Instead, it is more appropriate to consider a mixture of experts selected based on priors.** In Chapter 5, we propose a mechanism to explore representative expert models trained on typical learnable tasks, and the combined recorded expert models form a powerful prior known as the *foundation prior*. We also design an algorithm to utilize the foundation prior efficiently and effectively.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

This dissertation aims to understand the space of the machine learning tasks. Why is it important to investigate? Deep learning has achieved remarkable success in recent years. The AI field has flourished with many concepts and advancements (e.g., foundation models and multi-modality). In the past, researchers primarily regarded deep learning or machine learning algorithms as techniques for building models tailored to solve specific tasks. However, the current trend is to not only excel in a single task but also to build models that can handle multiple diverse tasks. This represents a significant shift in thinking over the past few years.

Furthermore, notable success has been achieved through this approach. Researchers have observed that language models (e.g., GPT) achieve impressive performance when trained on vast and diverse datasets. Integrating vision and language in multi-modality models (e.g., CLIP) is particularly powerful. However, based on statistical learning theory, we understand that training a model on multiple tasks does not guarantee a desired model that performs well on all given tasks. Nevertheless, researchers have developed models that can be fine-tuned and adapted to many diverse tasks in practice. This thesis seeks to address why machine learning can be effectively applied to a wide range of tasks. Understanding the space of tasks lays the foundation for answering this question.

In this thesis, we define a task as a joint probability distribution of inputs and labels, denoted as $p(x, y)$. We focus on researching the properties of *typical learning tasks*. It is important to differentiate between typical learning tasks and the entire set of all tasks. The set of all tasks encompasses arbitrary joint distributions that are highly complex. Instead, our interest lies in investigating the space of typical learning tasks encountered in practical scenarios. To clarify, we define typical tasks as those currently addressed by researchers, such as image classification, image captioning, and reconstruction. Each individual researcher addresses a specific task, and the collection of these tasks forms a set of typical tasks. It is evident that large models trained on these typical tasks collectively achieve impressive results. Therefore, our objective is to gain a deeper understanding of this space of typical tasks.

If we aim to understand the space of tasks — particularly the joint probability distributions of typical task data, we can leverage information geometry—a well-established set of tools designed precisely for this purpose. The choice of information geometry stems from its explicit focus on understanding the geometric properties of probability distributions. Having defined our approach, next, we present our contributions cohesively.

In Chapter 2, we formalize the free energy principle that addresses how to perform pre-training effectively. The Information Bottleneck (IB) principle defines a minimal sufficient statistic of the data, proposing a representation that discards information not correlated with predicting labels. While such a representation is unique to the chosen task, it may perform poorly in predicting other labels that are correlated with the discarded information. On the other hand, if the representation contains redundant information about the data, it has the potential to predict other labels correlated with this extra information. We extend the concept of the information bottleneck and propose the notion of a world representation that adheres to the *free energy principle*. **The free energy principle identifies reconstruction as the canonical task that pre-training procedures should consider to preserve information and improve the representation quality for multiple other tasks.** Today, reconstruction has become a successful mechanism for pre-training models, such as language models (e.g., BERT) trained to reconstruct the next token in an auto-regressive fashion. Leveraging insights from variational inference, the free energy principle foretold the effectiveness of reconstruction pre-training prior to its widespread adoption. Representations learned through this principle exhibit remarkable transferability, allowing for flexible adaptation to new tasks. Our experiments provide evidence of the effectiveness of the algorithm. This result is published in Gao and Chaudhari (2020b).

A representation that adheres to the free energy principle preserves the additional information and models the data-generating process aligned with the pre-training source task, denoted as $p(x, y)$. To better transfer such a representation to adapt to a new target task $p^{\text{new}}(x, y)$, it requires us to navigate the tasks from $p(x, y)$ to $p^{\text{new}}(x, y)$ properly in the space of the tasks. This serves as our motivation for our second contribution Chapter 3. By leveraging optimal transportation (OT), we establish a sequence of interpolated tasks that evolves from $p(x, y)$ to $p^{\text{new}}(x, y)$. The representation is then updated to align with the evolving data distribution. We refer to this process as *optimal coupled transfer*. Optimal coupled transfer facilitates model transfer, surpassing the direct fine-tuning approach on the target task. It enables the pre-trained model to traverse the shortest path in the space of tasks, thereby adapting to the new task efficiently. **From an information geometric perspective, the length of this shortest path connecting two tasks gives rise to a unique definition of the distance between them.** Consequently, we address a longstanding open question: how to define the distance between tasks theoretically soundly. We provide experimental evidence to support our viewpoints. Through minor modifications in the code, we update models to adapt to the sequential interpolated tasks. The results outperform the fine-tuning approach. These findings have been published in Gao and Chaudhari (2021).

The free energy principle highlights the effectiveness of reconstruction in pre-training. In the context of learning with unlabeled data $p(x)$, we can also harness the power of reconstruction. Successful algorithms in self-supervised learning (e.g., SimCLR) intuitively design the tasks to pre-train the models using the unlabelled data (e.g., representations invariant to the data augmentations) prior to knowing the actual downstream task $p(x, y)$. **Instead of artificially designing tasks, a more canonical approach to explore unlabeled data is to directly reconstruct the potential downstream tasks $p(x, y)$ in the space of tasks.** We frame this question as the choice of the prior in Bayesian statistics Chapter 4. By leveraging the power of a *reference prior*, we reconstruct a pool of diverse tasks that encompass the typical downstream tasks without knowing the actual labels. Through empirical experiments, we demonstrate the effectiveness of our approach, achieving state-of-the-art results. These findings have been published in Gao et al. (2022).

Over the past year, we have observed that our previous results can be formalized as constructions in the *prediction space*. Chapter 5 introduces an information-geometric technique for analyzing the probabilistic models underlying deep neural networks. We present key information geometric concepts, including prediction space, divergence, infinitesimal distance, and visualization methods.

Utilizing this new language allows us to interpret our previous results more simply and elegantly. Currently, many researchers are pursuing the development of foundation models. However, statistical learning theory insights suggest that building a single model for all tasks may not be ideal. **Instead, it is more appropriate to consider a mixture of experts selected based on priors, as opposed to relying solely on an overconfident point estimator.** In Chapter 5, we propose a mechanism to explore representative expert models trained on typical learnable tasks, and the combined recorded models form a powerful prior known as the *foundation prior*. We also design an algorithm to utilize the foundation prior efficiently, and our experimental results demonstrate the algorithm's effectiveness. It is important to note that while foundation models may not be suitable for all tasks, the foundation prior, formally a mixture of experts, is expected to perform better.

CHAPTER 2

INFORMATION THEORETICAL PRINCIPLES FOR REPRESENTATION LEARNING

This chapter presents an information-theoretic principle for pre-training representations that aligns with various existing information criteria in machine learning. We established a formal connection between this framework and the free energy principle in physics, highlighting the relationship and effectiveness.

2.1. Information Bottleneck Principle for Representation Learning

The *information bottleneck method* is a technique in information theory introduced by (Tishby et al., 2000). Given a joint probability distribution $p(x, y)$ between input data x and an observed relevant label y. The information bottleneck method is designed to find the best trade-off between classification accuracy and compression complexity of the data representation z.

The information bottleneck can also be viewed as a rate-distortion problem, with a distortion function that measures how well the label y is predicted from a compressed representation z compared to its direct prediction from input x. This interpretation provides a general algorithm for solving the information bottleneck trade-off and calculating the information curve from the given joint distribution $p(x, y)$. Let $e(z|x)$ denote an encoder that encodes input x into a latent representation z. Let $c(y|z)$ denote a classifier that predicts y given a representation z. The information bottleneck method minimizes the following functions:

$$\min_{e(z|x),c(y|z)} I(x\ ;\ z) - \lambda I(z\ ;\ y). \tag{2.1}$$

where $I(x\ ;\ z)$ and $I(z\ ;\ y)$ are the mutual information of x and z, and of z and y, respectively, and λ is a Lagrange multiplier.

The information bottleneck generalized the classical notion of minimal sufficient statistics from parametric statistics to arbitrary distributions. The representation z is useful to predict the correct label y by maximizing the mutual information $I(z\ ;\ y)$. Such a representation z is thus a statistic of the data *sufficient* for the task of classification. While z is also *minimal*—say in its size—it would discard information in the data that is not correlated with the labels by minimizing $I(x\ ;\ z)$.

Information Bottleneck is used to open the black box of Deep Neural Networks (DNN). Let z be any hidden layer of the network. (Shwartz-Ziv and Tishby, 2017) proposed the information bottleneck that expresses the trade-off between the mutual information measures $I(x\ ;\ z)$ and $I(z\ ;\ y)$. In this case, $I(x\ ;\ z)$ and $I(z\ ;\ y)$ respectively quantify the amount of information that the hidden layer contains about the input and the output. They conjectured that the training process of a DNN consists of two separate phases; 1) an initial fitting phase in which $I(z\ ;\ y)$ increases, and 2) a subsequent compression phase in which $I(x\ ;\ z)$ decreases. (Saxe et al., 2019) countered the claim of (Shwartz-Ziv and Tishby, 2017), stating that this compression phenomenon in DNNs is not comprehensive and it depends on the particular activation function. In particular, they claimed that compression does not happen with ReLu activation functions. However, (Noshad et al., 2019) used a rate-optimal estimator of mutual information to explore this controversy, observing that the optimal hash-based estimator reveals the compression phenomenon in a wider range of networks with ReLu and max-pooling activation. Recently, (Goldfeld et al., 2018; Geiger, 2021) argue that the observed compression is a result of geometric instead of information-theoretic phenomena.

2.2. Free Energy Principle for Representation Learning

The minimal sufficient representation z in (2.1) is unique to the chosen task $p(x, y)$. It discards information not correlated with predicting labels and may perform poorly in predicting other labels that are correlated with the discarded information. In addition, this representation in (2.1) does not offer a complete modeling of the real-world data generating process $p(x, y)$–it completely ignores information of $p(x)$.

If, instead, the representation were to model the world and have lots of redundant information about the data, it could potentially predict other labels correlated with this extra information. Therefore, we are primarily interested in learning a *world representation* z that encodes and compress the information from the real-world data-generating process $p(x, y)$, based on a fundamental assumption.

We will denote expectation over data using the notation $\langle\varphi\rangle_{p(x)} = \int \mathrm{d}x\ p(x)\varphi$.

Assumption 1. We accept a traditional variational inference modeling assumption, in which representation z acts as a *latent factor* for generating x and y, rendering them conditionally independent, leaving no unexplained correlations. We assume the latent factor z is generated from a margin $\pi(z)$. Then a decoder d decodes z back into the original data x, and a classifier $c(y|z)$ creates label y from z. Let $e(z|x)$ denote an encoder that encodes data x into a latent code z. $e(z|x)$, $d(x|z)$, together with $c(y|z)$ model the data generating process $q(x, y)$ satisfying the Bayesian rule,

$$q(x, y)e(z|x) = m(z)d(x|z)c(y|z). \tag{2.2}$$

We align the modeling data-generating process $q(x, y)$ (2.2) with the real data process $p(x, y)$ by minimizing the cross-entropy loss,

$$\begin{aligned}
-\mathbb{E}_{x,y\sim p(x,y)} \log q(x, y) &= \mathbb{E}_{x,y\sim p(x,y)} [\log e(z|x) - \log m(z) - \log d(x|z) - \log c(y|z)] \\
&= \mathbb{E}_{x,y\sim p(x,y)} \left[\int \mathrm{d}z\ e(z|x) \left(\log \frac{e(z|x)}{m(z)} - \log d(x|z) - \log c(y|z)\right)\right] \\
&= \mathbb{E}_{x\sim p(x)} \left[\int \mathrm{d}z\ e(z|x) \log \frac{e(z|x)}{m(z)}\right] + \mathbb{E}_{x\sim p(x)} \left[-\int \mathrm{d}z\ e(z|x) \log d(x|z)\right] \\
&\quad + \mathbb{E}_{x,y\sim p(x,y)} \left[-\int \mathrm{d}z\ e(z|x) \log c(y|z)\right] \\
&= R + D + C.
\end{aligned} \tag{2.3}$$

The rate

$$R := \mathbb{E}_{x\sim p(x)} \left[\int \mathrm{d}z\ e(z|x) \log \frac{e(z|x)}{m(z)}\right] \tag{2.4}$$

is a Kullback-Leibler (KL) divergence; it measures the average excess bits used to encode samples from $e(z|x)$ using a code that was built for our approximation of the true marginal on the latent factors $m(z)$. The distortion

$$D := \mathbb{E}_{x\sim p(x)} \left[-\int \mathrm{d}z\ e(z|x) \log d(x|z)\right] \tag{2.5}$$

measures the quality of reconstruction of the decoder $d(x|z)$ and

$$C = \mathbb{E}_{x,y\sim p(x,y)}\left[-\int \mathrm{d}z\ e(z|x)\log c(y|z)\right] \tag{2.6}$$

measures the classification loss of the classifier $c(y|z)$. This also leads to the study of the following Lagrangian, which is similar to the Information Bottleneck of Tishby et al. (2000),

$$F(\lambda,\gamma) = \min_{e(z|x),d(x|z),c(y|z)} R + \lambda D + \gamma C. \tag{2.7}$$

As Alemi and Fischer (2018) show, this Lagrangian can be formally connected to ideas in thermodynamics. We will heavily exploit and specialize this point of view. In particular, the objective $F(\lambda,\gamma)$ can be rewritten as maximizing the log-partition function (2.14), also known as the *free-energy* in statistical physics (Mezard and Montanari, 2009). (2.7) extends the concept of the information bottleneck and defines a mechanism for pre-training representations. We refer to this as the *free energy principle* for representation learning.

The rate R (2.4) and the classification loss C (2.6) are the building blocks of the deep variational information bottleneck (Alemi et al., 2016). The cool part is the **free energy principle identifies reconstruction D (2.5) as the canonical task that pre-training procedures should consider to preserve information and improve the representation quality.** Today, reconstruction has become a successful mechanism for pre-training models, such as language models (e.g., BERT) trained to reconstruct the next token in an auto-regressive fashion. Leveraging insights from variational inference (2.2), the free energy principle foretold the effectiveness of reconstruction pre-training prior to its widespread adoption.

2.2.1. Rate-Distortion curve

Let

$$H := \mathbb{E}_{x\sim p(x)}\left[-\log p(x)\right]$$

denote the Shanon entropy of the true data distribution; it quantifies the complexity of the data. The functionals R, D in (2.2), and H come together to give the inequality,

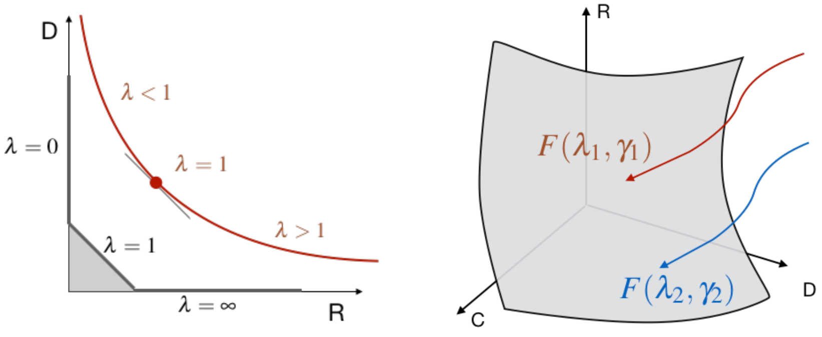
$$H - D \leq I_e(x; z) \leq R \tag{2.8}$$

where mutual information $I_e = \mathbb{E}_{x\sim p(x)}\mathrm{KL}(e(z|x) \mid\mid m(z))$ is the KL-divergence between the learned encoder and the true (unknown) conditional of the latent factors.

The outer inequality $H \leq D + R$ forms the basis for a large body of literature on Evidence Lower Bounds (ELBO, see Kingma and Welling (2013)). Consider Fig. 2.1a, if the capacity of our candidate distributions $e(z|x), m(z)$ and $d(x|z)$ is infinite, we can obtain the equality $H = R + D$. This is the thick black line in Fig. 2.1a.

For finite capacity variational families, say parameterized by θ, which we denote by $e_\theta(z|x)$, $d_\theta(x|z)$ and $m_\theta(z)$ respectively, as Alemi et al. (2017) argue, one obtains a convex RD curve (shown in red in Fig. 2.1a) corresponding to the Lagrangian

$$F(\lambda) = \min_{e_\theta(z|x), m_\theta(z), d_\theta(x|z)} R + \lambda D. \tag{2.9}$$



(a)

(b)

Figure 2.1: **Schematic of the equilibrium surface.** Fig. 2.1a shows that rate (R) and distortion (D) trade off against each other on the equilibrium surface. Similarly in Fig. 2.1b, the equilibrium surface is a convex constraint that joins rate, distortion and the classification loss. Training objectives with different (λ, γ) (shown in red and blue) reach different parts of the equilibrium surface.

This Lagrangian is the relaxation of the idea that given a fixed variational family and data distribution $p(x)$, there exists an optimal value of, say, rate $R = f(D)$ that best sandwiches (2.8). The optimal Lagrange multiplier is $\lambda = \frac{\partial R}{\partial D}$ evaluated at the desired value of D.

2.2.2. Rate-Distortion-Classification surface

If the parameters of the model—which now consists of the encoder $e(z|x)$, decoder $d(x|z)$ and the classifier $c(y|z)$—are denoted by θ, the training process for the model ($\min_\theta R + \lambda D + \gamma C$) induces a distribution $p(\theta| \{(x, y)\})$ where $\{(x, y)\}$ denotes a finite dataset. In addition to R, D and C, the authors in Alemi and Fischer (2018) define

$$S = \mathbb{E}_{x \sim p(x), y \sim p(y|x)} \left[\log \frac{p(\theta| \{x, y\})}{m(\theta)} \right] \tag{2.10}$$

which is the relative entropy of the distribution on parameters θ after training compared to a prior distribution $m(\theta)$ of our choosing. Using a very similar argument as Sec. 2.2.1 the four functionals R, D, C and S form a convex three-dimensional surface in the RDCS phase space. A schematic is shown in Fig. 2.1b for $\sigma = 0$. We can again consider a Lagrange relaxation of this surface given by

$$F(\lambda, \gamma, \sigma) = \min_{e(z|x), m(z), d(x|z), c(y|z)} R + \lambda D + \gamma C + \sigma S. \tag{2.11}$$

Remark 2 (‘The ‘First Law” of learning). Alemi and Fischer (2018) draw formal connections of the Lagrangian in (2.11) with the theory of thermodynamics. Just like the first law of thermodynamics is a statement about the conservation of energy in physical processes, the fact that the four functionals are tied together in a smooth constraint $f(R, D, C, S) = 0$ leads to an equation of the form

$$\mathrm{d}R = -\lambda \ \mathrm{d}D - \gamma \ \mathrm{d}C - \sigma \ \mathrm{d}S \tag{2.12}$$

which indicates that information in learning processes is conserved. The information in the latent representation z is kept either to reconstruct back the original data or to predict the labels. The former is captured by the encoder-decoder pair, the latter is captured by the classifier.

Remark 3 (Setting $\sigma = 0$). The distribution $p(\theta| \{(x, y)\})$ is a posterior on the parameters of the model given the dataset. While this distribution is well-defined under minor technical conditions, e.g., ergodicity, performing computations with this distribution is difficult. **We therefore only consider the case when $\sigma = 0$ in the sequel** and leave the general case for future work.

The following lemma (proved in Sec. 2.6.2) shows that the constraint surface connecting the information-theoretic functionals R, D and C is convex and its dual, the Lagrangian $F(\lambda, \gamma)$ is concave.

Lemma 4 (The RDC constraint surface is convex). The constraint surface $f(R, D, C) = 0$ is convex and the Lagrangian $F(\lambda, \gamma)$ is concave.

We can show using a similar proof that the entire surface joining R, D, C and S is convex by considering the cases $\lambda = 0$ and $\gamma = 0$ separately. Note that the constraint is convex in R, D and C; it need not be convex in the model parameters θ that parameterize $e_\theta(z|x), m_\theta(z)$, etc.

2.2.3. **Equilibrium surface of optimal free-energy**

We next elaborate upon the objective in (2.11). Consider the functionals R, D and C parameterized using parameters $\theta \in \Theta \subseteq \mathbb{R}^N$. First, consider the problem

$$F(\lambda, \gamma) = \min_{e(z|x),\ \theta \in \Theta} \quad R + \lambda D + \gamma C. \tag{2.13}$$

We can solve this using calculus of variations to get the optimal encoder,

$$e(z|x) \propto m_\theta(z)\ d_\theta(x|z)^\lambda \exp\left(\gamma \int\ \mathrm{d}y\ p(y|x)\ \log c_\theta(y|z)\right).$$

We assume in this paper that the labels are a deterministic function of the data, i.e., $p(y|x) = \delta(y - y_x)$ where y_x is the true label of the datum x. We therefore have

$$e(z|x) = \frac{m_\theta(z) d_\theta(x|z)^\lambda c_\theta(y_x|z)^\gamma}{Z_{\theta,x}}$$

where the normalization constant is

$$Z_{\theta,x} = \int \mathrm{d}z\ m_\theta(z) d_\theta(x|z)^\lambda c_\theta(y_x|z)^\gamma. \tag{2.14}$$

The objective $F(\lambda,\gamma)$ can now be rewritten as maximizing the log-partition function, also known as the free-energy in statistical physics (Mezard and Montanari, 2009),

$$F(\lambda,\gamma) = \min_{\theta\in\Theta}\ -\langle \log Z_{\theta,x}\rangle_{p(x)}. \tag{2.15}$$

Remark 5 (Why is it called the "equilibrium" surface?). Given a finite dataset $\{(x,y)\}$, one may minimize the objective in (2.13) using stochastic gradient descent (SGD, Robbins and Monro (1951)) on a Hamiltonian

$$H(z;x,\theta,\lambda,\gamma) \equiv -\log m_\theta(z) - \lambda \log d_\theta(x|z) - \gamma \log c_\theta(y|z) \tag{2.16}$$

with updates given by

$$\theta^{k+1} = \theta^k - \sigma\ \nabla_\theta\, \mathbb{E}_{x\sim p(x)} \left[\int \mathrm{d}z\ e_{\theta^k}(z|x) H(z;x,\theta^k,\lambda,\gamma) \right] \tag{2.17}$$

where $\sigma > 0$ is the step-size; the gradient ∇_θ is evaluated over samples from $p(x)$ and $e_\theta(z|x)$. Using the same technique as that of Chaudhari and Soatto (2017), one can show that the objective

$$\mathbb{E}_{\theta\sim p(\theta|\{x,y\})}\ \left[\langle -\log Z_{\theta,x}\rangle_{p(x)} \right] - \sigma H(p(\theta\ |\ \{x,y\})).$$

decreases *monotonically*. Observe that our objective in (2.13) corresponds to the limit $\sigma \to 0$ of this objective along with a uniform non-informative prior $m(\theta)$ in (2.10). In fact, this result is analogous to the classical result that an ergodic Markov chain makes monotonic improvements in the KL-divergence as it converges to the steady-state, also known as, equilibrium, distribution (Levin and Peres, 2017). The posterior distribution of the model parameters induced by the stochastic updates in (2.17) is the Gibbs distribution $p^*(\theta\ |\ \{(x,y)\}) \propto \exp\left(-2(R + \lambda D + \gamma C)/\sigma\right)$.

It is for the above reason that we call the surface in Fig. 2.1b parameterized by

$$\Theta_{\lambda,\gamma} = \left\{ \theta \in \Theta : - \left\langle \log Z_{\theta,x} \right\rangle_{p(x)} = F(\lambda,\gamma) \right\} \tag{2.18}$$

as the "equilibrium surface". Learning, in this case minimizing (2.13), is initialized outside this surface and converges to specific parts of the equilibrium surface depending upon (λ,γ); this is denoted by the red and blue curves in Fig. 2.1b. The constraint that ties results in this equilibrium surface is that variational inequalities such as (2.8) (more are given in Alemi and Fischer (2018)) are tight up to the capacity of the model. This is analogous to the concept of equilibrium in thermodynamics (Sethna, 2006)

2.3. Dynamical Processes on the Equilibrium Surface

This section prescribes dynamical processes that explore the equilibrium surface. For any parameters $\theta \in \Theta$, not necessarily on the equilibrium surface, let us define

$$J(\theta,\lambda,\gamma) = - \left\langle \log Z_{\theta,x} \right\rangle_{p(x)}. \tag{2.19}$$

If $\theta \in \Theta_{\lambda,\gamma}$ we have $J(\theta,\lambda,\gamma) = F(\lambda,\gamma)$ which implies

$$\nabla_\theta J(\theta,\lambda,\gamma) = 0 \text{ for all } \theta \in \Theta_{\lambda,\gamma}. \tag{2.20}$$

Quasi-static process. A quasi-static process in thermodynamics happens slowly enough for a system to remain in equilibrium with its surroundings. In our case, we are interested in evolving Lagrange multipliers (λ,γ) slowly and simultaneously keep the model parameters θ on the equilibrium surface; the constraint (2.20) thus holds at each time instant. The equilibrium surface is parameterized by R, D and C so changing (λ,γ) adapts the three functionals to track their optimal values corresponding to $F(\lambda,\gamma)$.

Let us choose some values $(\dot\lambda,\dot\gamma)$ and the trivial dynamics $\frac{\mathrm{d}}{\mathrm{d}t}\lambda = \dot\lambda$ and $\frac{\mathrm{d}}{\mathrm{d}t}\gamma = \dot\gamma$. The quasi-static constraint leads to the following partial differential equation (PDE)

$$0 \equiv \frac{\mathrm{d}}{\mathrm{d}t} \nabla_\theta J(\theta, \lambda, \gamma) = \nabla_\theta^2 J \, \dot{\theta} + \dot{\lambda} \frac{\partial}{\partial \lambda} \nabla_\theta J + \dot{\gamma} \frac{\partial}{\partial \gamma} \nabla_\theta J \tag{2.21}$$

valid all $\theta \in \Theta_{\lambda,\gamma}$. At each location $\theta \in \Theta_{\lambda,\gamma}$ the above PDE indicates how the parameters should evolve upon changing the Lagrange multipliers (λ, γ). We can rewrite the PDE using the Hamiltonian H in (2.16) as shown next.

Lemma 6 (Equilibrium dynamics for parameters θ). Given $(\dot{\lambda}, \dot{\gamma})$, the parameters $\theta \in \Theta_{\lambda,\gamma}$ evolve as

$$\begin{aligned} \dot{\theta} &= A^{-1} b_\lambda \; \dot{\lambda} + A^{-1} b_\gamma \; \dot{\gamma} \\ &= \theta_\lambda \dot{\lambda} + \theta_\gamma \dot{\gamma} \end{aligned} \tag{2.22}$$

where H is the Hamiltonian in (2.16) and

$$\begin{aligned} A &= \nabla_\theta^2 J = \mathbb{E}_{x \sim p(x)} \left[\left\langle \nabla_\theta^2 H \right\rangle + \left\langle \nabla_\theta H \right\rangle \left\langle \nabla_\theta H \right\rangle^\top - \left\langle \nabla_\theta H \; \nabla_\theta^\top H \right\rangle \right]; \\ b_\lambda &= -\frac{\partial}{\partial \lambda} \nabla_\theta J = -\mathbb{E}_{x \sim p(x)} \left[\left\langle \frac{\partial \nabla_\theta H}{\partial \lambda} \right\rangle - \left\langle \frac{\partial H}{\partial \lambda} \nabla_\theta H \right\rangle + \left\langle \frac{\partial H}{\partial \lambda} \right\rangle \left\langle \nabla_\theta H \right\rangle \right]; \\ b_\gamma &= -\frac{\partial}{\partial \gamma} \nabla_\theta J = -\mathbb{E}_{x \sim p(x)} \left[\left\langle \frac{\partial \nabla_\theta H}{\partial \gamma} \right\rangle - \left\langle \frac{\partial H}{\partial \gamma} \nabla_\theta H \right\rangle + \left\langle \frac{\partial H}{\partial \gamma} \right\rangle \left\langle \nabla_\theta H \right\rangle \right]. \end{aligned}$$

All the inner expectations $\langle \cdot \rangle$ above are taken with respect to the Gibbs measure of the Hamiltonian, i.e., $\langle \varphi \rangle = \frac{\int \varphi \exp(-H(z)) \, \mathrm{d}z}{\int \exp(-H(z)) \, \mathrm{d}z}$. The dynamics for the parameters θ is therefore a function of the two directional derivatives

$$\theta_\lambda = A^{-1} \, b_\lambda, \quad \text{and} \quad \theta_\gamma = A^{-1} \, b_\gamma \tag{2.23}$$

with respect to λ and γ. Note that A in (2.22) is the Hessian of a strictly convex functional.

This lemma allows us to implement dynamical processes for the model parameters θ on the equilibrium surface. As expected, this is an ordinary differential equation (2.22) that depends on our chosen evolution for $(\dot{\lambda}, \dot{\gamma})$ through the directional derivatives $\theta_\lambda, \theta_\gamma$. The utility of the above lemma therefore lies in the expressions for these directional derivatives. Sec. 2.6.3 gives the proof of the above lemma.

Remark 7 (Implementing the equilibrium dynamics). The equations in Lemma 6 may seem complicated to compute but observe that they can be readily estimated using samples from the dataset $x \sim p(x)$ and those from the encoder $z \sim e_\theta(z|x)$. The key difference between (2.22) and, say, the ELBO objective is that the gradient in the former depends upon the Hessian of the Hamiltonian H. These equations can be implemented using Hessian-vector products (Pearlmutter, 1994). If the dynamics involves certain constrains among the functionals, as Rem. 8 shows, we simplify the implementation of such equations.

2.3.1. **Iso-classification process**

An iso-thermal process in thermodynamics is a quasi-static process where a system exchanges energy with its surroundings and remains in thermal equilibrium with the surroundings. We now analogously define an iso-classification process that adapts parameters of the model θ while the free-energy is subject to slow changes in (λ, γ). This adaptation is such that the classification loss is kept constant while the rate and distortion change automatically.

Following the development in Lemma 6, it is easy to create an iso-classification process. We simply add a constraint of the form

$$\begin{aligned} \frac{\mathrm{d}}{\mathrm{d}t} \nabla_\theta J &= 0 \qquad \text{(Quasi-Static Condition)} \\ \frac{\mathrm{d}}{\mathrm{d}t} C &= 0 \qquad \text{(Iso-classification Condition).} \end{aligned} \tag{2.24}$$

Using a very similar computation (given in Sec. 2.6.4) as that in the proof of Lemma 6, this leads to the constrained dynamics

$$\begin{aligned} 0 &= C_\lambda \dot{\lambda} + C_\gamma \dot{\gamma} \\ \dot{\theta} &= \theta_\lambda \dot{\lambda} + \theta_\gamma \dot{\gamma}. \end{aligned} \tag{2.25}$$

The quantities C_λ and C_γ are given by

$$\begin{aligned} C_\lambda &= -\mathbb{E}_{x \sim p(x)} \left[\left\langle \frac{\partial H}{\partial \lambda} \right\rangle \langle \ell \rangle - \left\langle \frac{\partial H}{\partial \lambda}\ \ell \right\rangle + \left\langle \theta_\lambda^\top \nabla_\theta\, H \right\rangle \langle \ell \rangle - \left\langle \ell \theta_\lambda^\top \nabla_\theta\, H \right\rangle + \left\langle \theta_\lambda^\top \nabla_\theta\, \ell \right\rangle \right] \\ C_\gamma &= -\mathbb{E}_{x \sim p(x)} \left[\left\langle \frac{\partial H}{\partial \gamma} \right\rangle \langle \ell \rangle - \left\langle \frac{\partial H}{\partial \gamma}\ \ell \right\rangle + \left\langle \theta_\gamma^\top \nabla_\theta\, H \right\rangle \langle \ell \rangle - \langle \ell \theta_\gamma^T \nabla_\theta\, H \rangle + \left\langle \theta_\gamma^\top \nabla_\theta\, \ell \right\rangle \right] \end{aligned} \tag{2.26}$$

where $\ell = \log c_\theta(y_x|z)$ is the logarithm of the classification loss. Observe that we are not free to pick any values for $(\dot{\lambda}, \dot{\gamma})$ for the iso-classification process anymore, the constraint $\frac{\mathrm{d}C}{\mathrm{d}t} = 0$ ties the two rates together.

Remark 8 (Implementing an iso-classification process). The first constraint in (2.25) allows us to choose

$$\begin{aligned} \dot{\lambda} &= -\alpha \frac{\partial C}{\partial \gamma} = -\alpha \frac{\partial^2 F}{\partial \gamma^2} \\ \dot{\gamma} &= \alpha \frac{\partial C}{\partial \lambda} = \alpha \frac{\partial^2 F}{\partial \lambda \partial \gamma} \end{aligned} \tag{2.27}$$

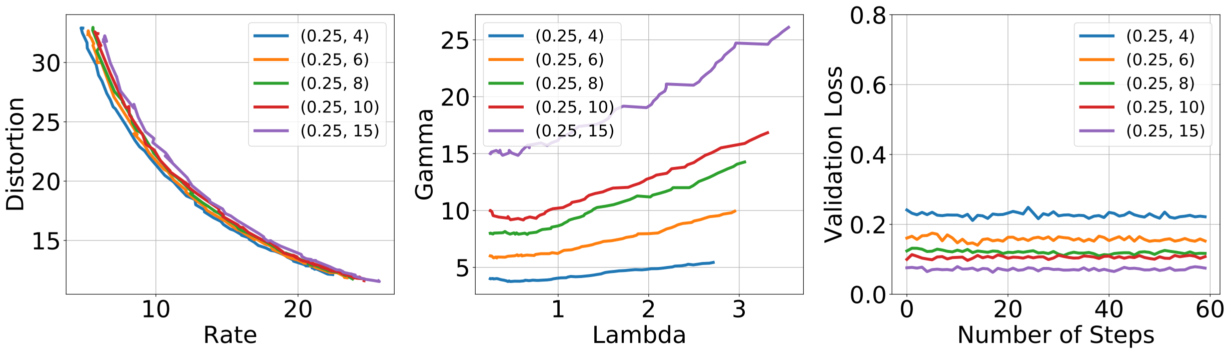
where α is a parameter to scale time. The second equalities in both rows follow because $F(\lambda, \gamma)$ is the optimal free-energy which implies relations like $D = \frac{\partial F}{\partial \lambda}$ and $C = \frac{\partial F}{\partial \gamma}$. We can now compute the two deriatives in (2.27) using finite differences to implement an iso-classification process. This is equivalent to running the dynamics in (2.25) using finite-difference approximation for the terms $\frac{\partial H}{\partial \lambda}, \frac{\partial H}{\partial \gamma}, \frac{\partial \nabla_\theta H}{\partial \lambda}, \frac{\partial \nabla_\theta H}{\partial \gamma}$. While approximating all these listed quantities at each update of θ would be cumbersome, exploiting the relations in (2.25) is efficient even for large neural networks, as our experiments show.

Remark 9 (Other dynamical processes of interest). In this chapter, we focus on iso-classification processes. However, following the same program as that of this section, we can also define other processes of interest, e.g., one that keeps $C + \beta^{-1} R$ constant while fine-tuning a model. This is similar to the alternative Information Bottleneck of Achille and Soatto (2017) wherein the rate is defined using the weights of a network as the random variable instead of the latent factors z. This is also easily seen to be the right-hand side of the PAC-Bayes generalization bound (McAllester, 2013). A dynamical process that preserves this functional would be able to control the generalization error which is an interesting prospect for future work.

2.3.2. Experimental validation: Iso-classification process on the equilibrium surface

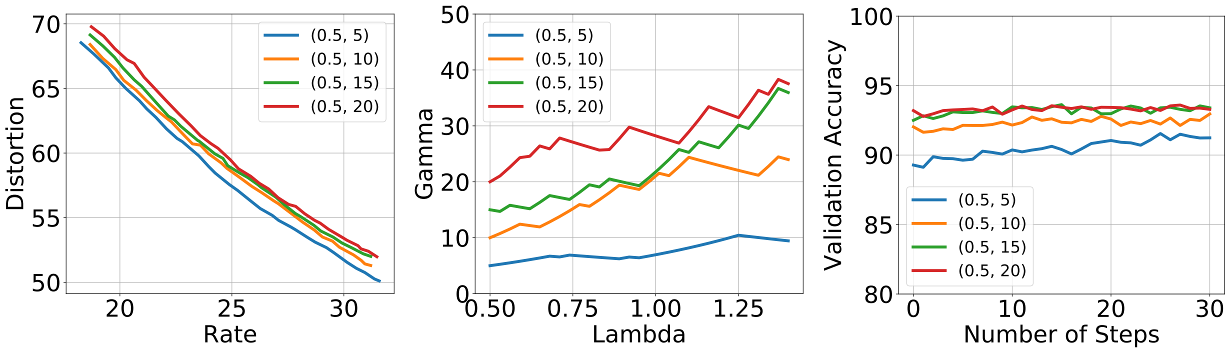
This section implements the dynamics in Sec. 2.3 that traverses the equilibrium surface.

Setup We use the MNIST (LeCun et al., 1998) and CIFAR-10 (Krizhevsky, 2009) datasets for our experiments. We use a 2-layer fully-connected network (same as that of Kingma and Welling



(a) (b) (c)

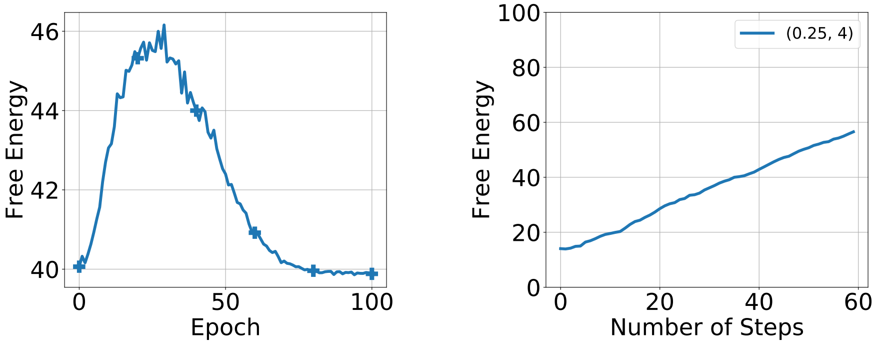
Figure 2.2: **Iso-classification process for MNIST.** We run 5 different experiments for initial Lagrange multipliers given by $\lambda = 0.25$ and $\gamma \in \{4, 6, 8, 10, 15\}$. During each experiment, we modify these Lagrange multipliers (Fig. 2.2b) to keep the classification loss constant and plot the rate-distortion curve (Fig. 2.2a) along with the validation loss (Fig. 2.2c). The validation accuracy is constant for each experiment; it is between 92–98% for these initial values of (λ, γ). Similarly the training loss is almost unchanged during each experiment and takes values between 0.06–0.2 for different values of (λ, γ).



(a) (b) (c)

Figure 2.3: **Iso-classification process for CIFAR-10.** We run 4 different experiments for initial Lagrange multipliers $\lambda = 0.5$ and $\gamma \in \{5, 10, 15, 20\}$. During each experiment, we modify the Lagrange multipliers (Fig. 2.3b) to keep the classification loss constant and plot the rate-distortion curve (Fig. 2.3a) along with the validation accuracy (Fig. 2.3c). The validation loss is constant during each experiment; it takes values between 0.5–0.8 for these initial values of (λ, γ). Similarly, the training loss is constant and takes values between 0.02–0.09 for these initial values of (λ, γ). Observe that the rate-distortion curve in Fig. 2.3a is much flatter than the one in Fig. 2.2a which indicates that the model family Θ for CIFAR-10 is much more powerful; this corresponds to the straight line in the RD curve for an infinite model capacity is as shown in Fig. 2.1a.

(2013)) as the encoder and decoder for MNIST; the encoder for CIFAR-10 is a ResNet-18 (He et al., 2016a) architecture while the decoder is a 4-layer deconvolutional network (Noh et al., 2015). Full details of the pre-processing, network architecture and training are provided in Sec. 2.6.1.



(a) (b)

Figure 2.4: **Variation of the free-energy $F(\lambda, \gamma)$ across the equilibration and the iso-classification processes.** Fig. 2.4a shows the free-energy during equilibration between small changes of (λ, γ). The initial and final values of the Lagrange multipliers are $(0.5, 1)$ and $(0.51, 1.04)$ respectively and the free-energy is about the same for these values. Fig. 2.4b shows the free-energy as (λ, γ) undergo a large change from their initial value of $(0.25, 4)$ to $(3.5, 26)$ during the iso-classification process in Fig. 2.2. Since the rate-distortion change a lot (Fig. 2.2a), the free-energy also changes a lot even if C is constant (Fig. 2.2c). Number of steps in Fig. 2.4b refers to the number of steps of running (2.28).

This experiment demonstrates the iso-classification process in Rem. 8. As discussed in Rem. 5, training a model to minimize the functional $R + \lambda D + \gamma C$ decreases the free-energy monotonically.

Details Given a value of the Lagrange multipliers (λ, γ) we first find a model on the equilibrium surface by training from scratch for 120 epochs with the Adam optimizer (Kingma and Ba, 2014); the learning rate is set to 10^{-3} and drops by a factor of 10 every 50 epochs. We then run the iso-classification process for these models in Rem. 8 as follows. We modify (λ, γ) according to the equations

$$\dot{\lambda} = -\alpha \frac{\partial C}{\partial \gamma} \quad \text{and} \quad \dot{\gamma} = \alpha \frac{\partial C}{\partial \lambda}. \tag{2.28}$$

Changes in (λ, γ) cause the equilibrium surface to change, so it is necessary to adapt the model parameters θ so as to keep them on the dynamically changing surface; let us call this process of adaptation "equilibriation". We achieve this by taking gradient-based updates to minimize $J(\lambda, \gamma)$ with a learning rate schedule that looks like a sharp quick increase from zero and then a slow annealing back to zero. The learning rate schedule is given by $\eta(t) = (t/T)^2 (1 - t/T)^5$ where t is the number of mini-batch updates taken since the last change in (λ, γ) and T is total number of mini-batch updates of equilibration. The maximum value of the learning rate is set to 1.5×10^{-3}. The free-energy should be unchanged if the model parameters are on the equilibrium surface after equilibration; this is shown in Fig. 2.4a. Partial derivatives in (2.28) are computed using finite-differences.

Fig. 2.2 shows the result for the iso-classification process for MNIST and Fig. 2.3 shows a similar result for CIFAR-10. We can see that the classification loss remains constant through the process. This experiment shows that we can implement an iso-classification process while keeping the model parameters on the equilibrium surface during it.

2.4. Transferring Representations to New Tasks

Sec. 2.3 demonstrated dynamical processes where the Lagrange multipliers λ, γ change with time and the process adapts the model parameters θ to remain on the equilibrium surface. This section demonstrates the same concept under a different kind of perturbation, namely the one where the underlying task changes. The prototypical example one should keep in mind in this section is that of transfer learning where a classifier trained on a dataset $p^s(x, y)$ is further trained on a new dataset, say $p^t(x, y)$. We will assume that the input domain of the two distributions is the same.

2.4.1. Changing the data distribution

If i.i.d samples from the source task are denoted by $X^s = \left\{x_1^s, \ldots, x_{n_s}^s\right\}$ and those of the target distribution are $X^t = \left\{x_1^t, \ldots, x_{n_t}^t\right\}$ the empirical source and target distributions can be written as

$$p^s(x) = \frac{1}{n_s} \sum_{i=1}^{n_s} \delta_{x - x_i^s}, \text{and } p^t(x) = \frac{1}{n_t} \sum_{i=1}^{n_t} \delta_{x - x_i^t}$$

respectively; here $\delta_{x-x'}$ is a Dirac delta distribution at x'. We will consider a transport problem that transports the source distribution $p^s(x)$ to the target distribution $p^t(x)$. For any $t \in [0, 1]$ we interpolate between the two distributions using a mixture

$$p(x, t) = (1 - t)p^s(x) + tp^t(x). \tag{2.29}$$

Observe that the interpolated data distribution equals the source and target distribution at $t = 0$ and $t = 1$ respectively and it is the mixture of the two distributions for other times. We keep the labels of the data the same and do not interpolate them. As discussed in Sec. 2.6.6 we can also use techniques from optimal transportation (Villani, 2008) to obtain a better transport; the same dynamical equations given below remain valid in that case.

2.4.2. **Iso-classification process with a changing data distribution**

The equilibrium surface $\Theta_{\lambda,\gamma}$ in Fig. 2.1b is a function of the task and also evolves with the task. We now give a dynamical process that keeps the model parameters in equilibrium as the task evolves quasi-statically. We again have the same conditions for the dynamics as those in (2.24). The following lemma is analogous to Lemma 6.

Lemma 10 (Dynamical process for changing data distribution). Given $(\dot{\lambda}, \dot{\gamma})$, the evolution of model parameters θ for a changing data distribution given by (2.29) is

$$\dot{\theta} = \theta_\lambda \dot{\lambda} + \theta_\gamma \dot{\gamma} + \theta_t \tag{2.30}$$

where

$$\theta_t = A^{-1}\ b_t =: -A^{-1} \int \frac{\partial p(x,t)}{\partial t}\ \langle \nabla_\theta\, H \rangle\ \mathrm{d}x \tag{2.31}$$

and the other quantities are as defined in Lemma 6 with the only change that expectations on data x are taken with respect to $p(x,t)$ instead of $p(x)$. The additional term θ_t arises because the data distribution changes with time.

A similar computation as that of Sec. 2.3.1 gives a quasi-static iso-classification process as the task evolves

$$\begin{aligned} \dot{\theta} &= \theta_\lambda \dot{\lambda} + \theta_\gamma \dot{\gamma} + \theta_t \\ 0 &= C_\lambda \dot{\lambda} + C_\gamma \dot{\gamma} + C_t \end{aligned} \tag{2.32}$$

where C_λ and C_γ are as given in (2.26) with the only change being that the outer expectation is taken with respect to $x \sim p(x,t)$. The new term that depends on time t is

$$C_t = -\int \frac{\partial p(x,t)}{\partial t} \langle \ell \rangle \, \mathrm{d}x - \mathbb{E}_{x \sim p(x,t)} \left[\left\langle \theta_t^\top \nabla_\theta H \right\rangle \langle \ell \rangle - \left\langle \theta_t^\top \nabla_\theta H \ \ell \right\rangle + \left\langle \theta_t^\top \nabla_\theta \ell \right\rangle \right] \tag{2.33}$$

with $\ell = \log c_\theta(y_{x_t}|z)$. Finally get

$$\begin{aligned} \dot{\theta} &= \left(\theta_\lambda - \frac{C_\lambda}{C_\gamma}\,\theta_\gamma\right)\dot{\lambda} + \left(\theta_t - \frac{C_t}{C_\gamma}\theta_\gamma\right) \\ &=: \hat{\theta}_\lambda \dot{\lambda} + \hat{\theta}_t \end{aligned}. \tag{2.34}$$

This indicates that $\theta = \theta(\lambda, t)$ is a surface parameterized by λ and t, equipped with a basis of tangent plane $(\hat{\theta}_\lambda, \hat{\theta}_t)$.

2.4.3. **Geodesic transfer of representations**

The dynamics of Lemma 10 is valid for any $(\dot{\lambda}, \dot{\gamma})$. We provide a locally optimal way to change (λ, γ) in this section.

Remark 11 (Rate-distortion trade-off). Note that

$$\begin{aligned} &\dot{C} = 0, \\ &\dot{D} = \frac{\partial D}{\partial \lambda}\dot{\lambda} + \frac{\partial D}{\partial \gamma}\dot{\gamma} = -\alpha \left(\frac{\partial^2 F}{\partial \lambda^2}\frac{\partial^2 F}{\partial \gamma^2} - \left(\frac{\partial^2 F}{\partial \lambda \partial \gamma}\right)^2 \right) = -\alpha \det\left(\mathrm{Hess}(F)\right), \\ &\dot{R} = \frac{\partial R}{\partial D}\dot{D} + \frac{\partial R}{\partial C}\dot{C} = -\lambda \dot{D}. \end{aligned} \tag{2.35}$$

The first equality is simply our iso-classification constraint. For $\alpha > 0$, the second one indicates that $\dot{D} < 0$ using Lemma 4 which shows that $0 \succ \mathrm{Hess}(F)$. This also gives $\dot{\lambda} > 0$ in (2.27). The third equality is a powerful observation: it indicates a trade-off between rate and distortion, if $\dot{D} < 0$ we have $\dot{R} > 0$. It also shows the geometric structure of the equilibrium surface by connecting $\dot{R}$ and $\dot{D}$ together, which we will exploit next.

Computing the functionals R, D and C during the iso-classification transfer presents us with a curve in RDC space. Geodesic transfer implies that the functionals R, D follow the shortest path in this space. But notice that if **we assume that the model capacity is infinite**, the RDC space is Euclidean and therefore the geodesic is simply a straight line. Since we keep the classification loss constant during the transfer, $\dot{C} = 0$, straight line implies that slope $\mathrm{d}D/\mathrm{d}R$ is a constant, say k. Thus $\dot{D} = k\dot{R}$. Observe that $\dot{R} = \frac{\partial R}{\partial D}\dot{D} + \frac{\partial R}{\partial C}\dot{C} + \frac{\partial R}{\partial t} = -\lambda \dot{D} + \frac{\partial R}{\partial t}$. Combining the iso-classification constraint and the fact that $\dot{D} = k\dot{R} = -k\lambda\dot{D} + k\frac{\partial R}{\partial t}$, gives us a linear system:

$$\begin{aligned} &\frac{\partial D}{\partial t} + \frac{\partial D}{\partial \lambda}\dot{\lambda} + \frac{\partial D}{\partial \gamma}\dot{\gamma} = \frac{k\frac{\partial R}{\partial t}}{1+k\lambda}; \\ &\frac{\partial C}{\partial \lambda}\dot{\lambda} + \frac{\partial C}{\partial \gamma}\dot{\gamma} + \frac{\partial C}{\partial t} = 0 \end{aligned} \tag{2.36}$$

We solve this system to update (λ, γ) during the transfer.

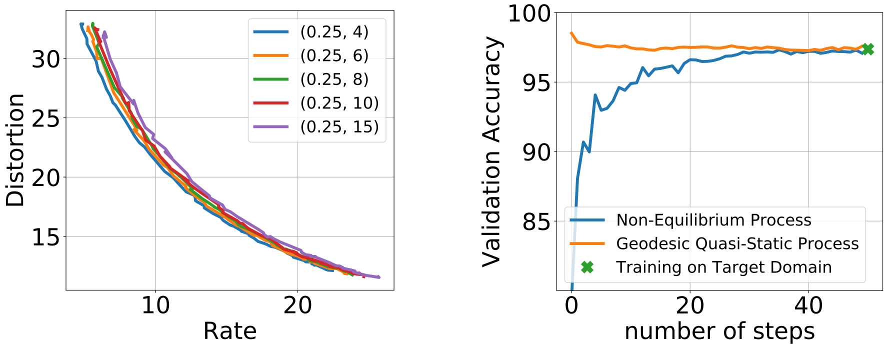
2.4.4. Experimental validation: transferring representations to new data

Representations learned through this principle exhibit remarkable transferability, allowing for flexible adaptation to new tasks. Our experiments provide evidence of the effectiveness of the algorithm. This section present experimental results of an iso-classification process for transferring the learnt representation.

Setup We use the MNIST (LeCun et al., 1998) and CIFAR-10 (Krizhevsky, 2009) datasets for our experiments. We use a 2-layer fully-connected network (same as that of Kingma and Welling (2013)) as the encoder and decoder for MNIST; the encoder for CIFAR-10 is a ResNet-18 (He et al., 2016a) architecture while the decoder is a 4-layer deconvolutional network (Noh et al., 2015). Full details of the pre-processing, network architecture and training are provided in Sec. 2.6.1.

We first pick the source dataset to be all images corresponding to digits 0–4 in MNIST and the target dataset is its complement, images of digits 5–9. Our goal is to adapt a model trained on the source task to the target task while keeping its classification loss constant. We run the geodesic transfer dynamics from Sec. 2.4.3 and the results are shown in Fig. 2.5.

It is evident that the classification accuracy is constant throughout the transfer and is also the same as that of training from scratch on the target. MNIST is an simple dataset and the accuracy gap



(a) (b)

Figure 2.5: **Transferring from source dataset of MNIST digits 0–4 to the target dataset consisting of digits 5–9.** Fig. 2.5a shows the variation of rate and distortion during the transfer; as discussed in Sec. 2.4.3 we maintain a constant $\mathrm{d}R/\mathrm{d}D$ during the transfer; the rate decreases and the distortion increases. Fig. 2.5b shows the validation accuracy during the transfer. The orange curve corresponds to geodesic iso-classification transfer; the blue curve is the result of directly fine-tuning the source model on the target data (note the very low accuracy at the start); the green point is the accuracy of training on the target task from scratch.

between iso-classification transfer, fine-tuning from the source and training from scratch is minor. The benefit of running the iso-classification transfer however is that we can be guaranteed about the final accuracy of the model. We expect the gap between these three to be significant for more complex datasets.

The iso-classification process is a quasi-static process, i.e., the model parameters θ are lie on the equilibrium surface at all times $t \in [0, 1]$ during the transfer. Note that both the equilibrium surface and the free-energy $F(\lambda, \gamma)$ are functions of the data and change with time. Let us write this explicitly as

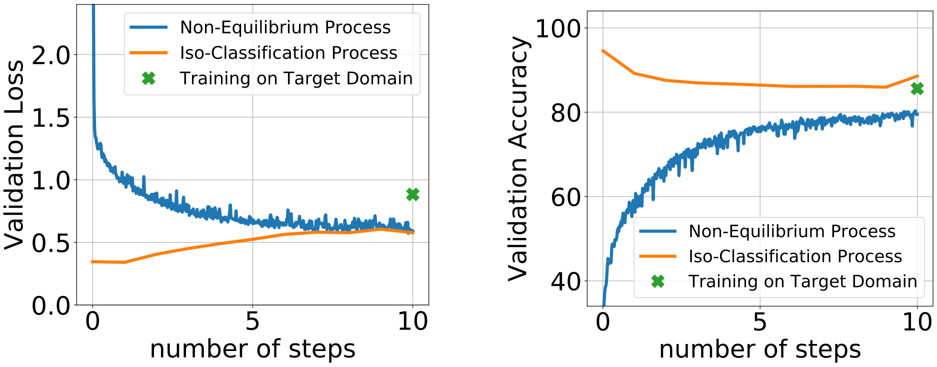
$$F(t) := R(t, \lambda(t), \gamma(t)) + \lambda D(t, \lambda(t), \gamma(t)) + \gamma C_0$$

where C_0 is the classification loss. We prescribed a geodesic transfer above where the Lagrange multipliers λ, γ were adapted simultaneously to confirm to the constraints of the equilibrium surface locally. We can forgot this and instead adapt them using the following heuristic. We let $\dot{\lambda} = k$ for some constant k and use

$$\frac{\partial C}{\partial \lambda}\dot{\lambda} + \frac{\partial C}{\partial \gamma}\dot{\gamma} + \frac{\partial C}{\partial t} = 0, \tag{2.37}$$

to get the evolution curve of $\gamma(t)$.

We next present experimental results of an iso-classification process for transferring the learnt representation. We pick the source dataset to be all vehicles (airplane, automobile, ship and truck) in CIFAR-10 and the target dataset consists of four animals (bird, cat, deer and dog). We set the output size of classifier to be four. Our goal is to adapt a model trained on the source task to the target task while keeping its classification loss constant. We run the iso-c transfer dynamics (2.37) and the results are shown in Fig. 2.6.



(a) (b)

Figure 2.6: **Transferring from source dataset of CIFAR-10 vehicles to the target dataset consisting of four animals.** Fig. 2.6a shows the variation of validation loss during the transfer. Fig. 2.6b shows the validation accuracy during the transfer. The orange curve corresponds to iso-classification transfer; the blue curve is the result of directly fine-tuning the source model on the target data (note the very low accuracy at the start); the green point is the accuracy of training on the target task from scratch.

It is evident that both the classification accuracy and loss are constant throughout the transfer. CIFAR-10 is a more complex dataset as comparing with MNIST and the accuracy gap between iso-classification transfer, fine-tuning from the source and training from scratch is significant. Observe that the classification loss gap between iso-classification transfer and training from scratch on the target is also significant. The benefit of running the iso-classification transfer is that we can be guaranteed about the final accuracy and validation loss of the model.

2.5. Related work and Discussion

We are motivated by the Information Bottleneck (IB) principle of Tishby et al. (2000); Shwartz-Ziv and Tishby (2017), which has been further explored by Achille and Soatto (2017); Alemi et al. (2016); Higgins et al.

(2017). The key difference in our work is that while these papers seek to understand the representation for a given task, we focus on how the representation can be adapted to a new task. Further, the Lagrangian in (2.13) has connections to PAC-Bayes bounds (McAllester, 2013; Dziugaite and Roy, 2017) and training algorithms that use the free-energy (Chaudhari et al., 2019). Our use of rate-distortion for transfer learning is close to the work on unsupervised learning of Brekelmans et al. (2019); Ver Steeg and Galstyan (2015).

This paper builds upon the work of Alemi et al. (2017); Alemi and Fischer (2018). We refine some results therein, viz., we provide a proof of the convexity of the equilibrium surface and identify it with the equilibrium distribution of SGD (Rem. 5). We introduce new ideas such as dynamical processes on the equilibrium surface. Our use of thermodynamics is purely as an inspiration; the work presented here is mathematically rigorous and also provides an immediate algorithmic realization of the ideas.

This paper has strong connections to works that study stochastic processes inspired from statistical physics for machine learning, e.g., approximate Bayesian inference and implicit regularization of SGD (Mandt et al., 2017; Chaudhari and Soatto, 2017), variational inference (Jordan et al., 1998; Kingma and Welling, 2013). The iso-classification process instantiates an "automatic" regularization via the trade-off between rate and distortion; this point-of-view is an exciting prospect for future work. The technical content of the paper also draws from optimal transportation (Villani, 2008).

A large number of applications begin with pre-trained models (Sharif Razavian et al., 2014; Girshick et al., 2014) or models trained on tasks different (Doersch and Zisserman, 2017). Current methods in transfer learning however do not come with guarantees over the performance on the target dataset, although there is a rich body of older work (Baxter, 2000) and ongoing work that studies this (Zamir et al., 2018). The information-theoretic understanding of transfer and the constrained dynamical processes developed in our paper is a first step towards building such guarantees. In this context, our theory can also be used to tackle catastrophic forgetting Kirkpatrick et al. (2017) to "detune" the model post-training and build up redundant features.

We presented dynamical processes that maintain the parameters of model on an equilibrium surface that arises out of a certain free-energy functional for the encoder-decoder-classifier architecture. The decoder acts as a measure of the information discarded by the encoder-classifier pair while fitting on a given task. We showed how one can develop an iso-classification process that travels on the equilibrium surface while keeping the classification loss constant. We showed an iso-classification transfer learning process which keeps the classification loss constant while adapting the learnt representation from a source task to a target task.

The information-theoretic point-of-view in this paper is rather abstract but its benefit lies in its exploitation of the equilibrium surface. Relationships between the three functionals, namely rate, distortion and classification, that define this surface, as also other functionals that connect to the capacity of the hypothesis class such as the entropy S may allow us to define invariants of the learning process. For complex models such as deep neural networks, such a program may lead an understanding of the principles that govern their working.

2.6. **Appendix**

2.6.1. Details of the experimental setup

Datasets. We use the MNIST (LeCun et al., 1998) and CIFAR-10 (Krizhevsky, 2009) datasets for these experiments. The former consists of 28 ×28-sized gray-scale images of handwritten digits (60,000 training and 10,000 validation). The latter consists of 32×32-sized RGB images (50,000 training and 10,000 for validation) spread across 10 classes; 4 of these classes (airplane, automobile, ship, truck) are transportation-based while the others are images of animals and birds.

Architecture and training. All models in our experiments consist of an encoder-decoder pair along with a classifier that takes in the latent representation as input. For experiments on MNIST, both encoder and decoder are multi-layer perceptrons with 2 fully-connected layers, the decoder uses a mean-square error loss, i.e., a Gaussian reconstruction likelihood and the classifier consists of a single fully-connected layer. For experiments on CIFAR-10, we use a residual network (He et al., 2016a) with 18 layers as an encoder and a decoder with one fully-connected layer and 4 deconvolutional layers (Noh et al., 2015). The classifier network for CIFAR-10 is a single fully-connected layer. All models use ReLU non-linearities and batch-normalization (Ioffe and Szegedy, 2015). Further details of the architecture are given in Sec. 2.6.1. We use Adam (Kingma and Ba, 2014) to train all models with cosine learning rate annealing.

The encoder and decoder for MNIST has 784–256–16 neurons on each layer; the encoding z is thus 16-dimensional which is the input to the decoder. The classifier has one hidden layer with 12 neurons and 10 outputs. The encoder for CIFAR-10 is a 18-layer residual neural network (ResNet-18) and the decoder has 4 deconvolutional layers. We used a slightly larger network for the geodesic transfer learning experiment on MNIST. The encoder and decoder have 784–400–64 neurons in each layer with a dropout of probability 0.1 after the hidden layer. The classifier has a single layer that takes the 64-dimensional encoding and predicts 10 classes.

2.6.2. Proof of Lemma 4

The second statement directly follows by observing that F is a minimum of affine functions in (λ, γ). To see the first, evaluate the Hessian of R and F

$$\operatorname{Hess}(R)\ \operatorname{Hess}(F) = \begin{pmatrix} \frac{\partial^2 R}{\partial D^2} & \frac{\partial^2 R}{\partial D \partial C} \\ \frac{\partial^2 R}{\partial C \partial D} & \frac{\partial^2 R}{\partial C^2} \end{pmatrix} \begin{pmatrix} \frac{\partial^2 F}{\partial \lambda^2} & \frac{\partial^2 F}{\partial \lambda \partial \gamma} \\ \frac{\partial^2 F}{\partial \gamma \partial \lambda} & \frac{\partial^2 F}{\partial \gamma^2} \end{pmatrix}$$

Since we have $F = \min_{e_\theta(z|x), d_\theta(x|z), m_\theta(z)} R + \lambda D + \gamma C$, we obtain

$$\lambda = -\frac{\partial R}{\partial D}, \quad \gamma = -\frac{\partial R}{\partial C}, \quad D = \frac{\partial F}{\partial \lambda}, \quad C = \frac{\partial F}{\partial \gamma}.$$

We then have

$$\begin{aligned} \mathrm{d}\lambda = -\mathrm{d}\left(\frac{\partial R}{\partial D}\right) &= -\frac{\partial^2 R}{\partial D^2}\, \mathrm{d}D - \frac{\partial^2 R}{\partial D \partial C}\, \mathrm{d}C \\ &= -\frac{\partial^2 R}{\partial D^2} \left(\frac{\partial D}{\partial \lambda} d\lambda + \frac{\partial D}{\partial \gamma} d\gamma\right) - \frac{\partial^2 R}{\partial D \partial C} \left(\frac{\partial C}{\partial \lambda} d\lambda + \frac{\partial C}{\partial \gamma} d\gamma\right) \\ &= -\left(\frac{\partial^2 R}{\partial D^2}\frac{\partial^2 F}{\partial \lambda^2} + \frac{\partial^2 R}{\partial D \partial C}\frac{\partial^2 F}{\partial \gamma \partial \lambda}\right) \mathrm{d}\lambda - \left(\frac{\partial^2 R}{\partial D^2}\frac{\partial^2 F}{\partial \lambda \partial \gamma} + \frac{\partial^2 R}{\partial D \partial C}\frac{\partial^2 F}{\partial \gamma^2}\right) \mathrm{d}\gamma; \end{aligned}$$

$$
\begin{aligned}
\mathrm{d}\gamma = -\mathrm{d}\left(\frac{\partial R}{\partial C}\right) &= -\frac{\partial^2 R}{\partial C \partial D}\ \mathrm{d}D - \frac{\partial^2 R}{\partial C^2}\ \mathrm{d}C \\
&= -\frac{\partial^2 R}{\partial C \partial D}\ \left(\frac{\partial D}{\partial \lambda} d\lambda + \frac{\partial D}{\partial \gamma} d\gamma\right) - \frac{\partial^2 R}{\partial C^2}\ \left(\frac{\partial C}{\partial \lambda} d\lambda + \frac{\partial C}{\partial \gamma} d\gamma\right) \\
&= -\left(\frac{\partial^2 R}{\partial C \partial D}\frac{\partial^2 F}{\partial \lambda^2} + \frac{\partial^2 R}{\partial C^2}\frac{\partial^2 F}{\partial \gamma \partial \lambda}\right)\ \mathrm{d}\lambda - \left(\frac{\partial^2 R}{\partial C \partial D}\frac{\partial^2 F}{\partial \lambda \partial \gamma} + \frac{\partial^2 R}{\partial C^2}\frac{\partial^2 F}{\partial \gamma^2}\right)\ \mathrm{d}\gamma.
\end{aligned}
$$

Compare the coefficients on both sides to get

$$
\begin{aligned}
\frac{\partial^2 R}{\partial D^2}\frac{\partial^2 F}{\partial \lambda^2} + \frac{\partial^2 R}{\partial D \partial C}\frac{\partial^2 F}{\partial \gamma \partial \lambda} &= \frac{\partial^2 R}{\partial C \partial D}\frac{\partial^2 F}{\partial \lambda \partial \gamma} + \frac{\partial^2 R}{\partial C^2}\frac{\partial^2 F}{\partial \gamma^2} = -1; \\
\frac{\partial^2 R}{\partial D^2}\frac{\partial^2 F}{\partial \lambda \partial \gamma} + \frac{\partial^2 R}{\partial D \partial C}\frac{\partial^2 F}{\partial \gamma^2} &= \frac{\partial^2 R}{\partial C \partial D}\frac{\partial^2 F}{\partial \lambda^2} + \frac{\partial^2 R}{\partial C^2}\frac{\partial^2 F}{\partial \gamma \partial \lambda} = 0,
\end{aligned}
$$

therefore

$$
\mathrm{Hess}(R)\ \mathrm{Hess}(F) = -I.
$$

Since $0 \succ \mathrm{Hess}(F)$, we have that $\mathrm{Hess}(R) \succ 0$, then the constraint surface $f(R, D, C) = 0$ is convex.

2.6.3. Proof of Lemma 6

Recall the definition of the objective function (2.19), first we compute the gradient of the objective function as following:

$$
\begin{aligned}
\nabla_\theta\, J(\theta, \lambda, \gamma) &= -\mathbb{E}_{x\sim p(x)}\, \nabla_\theta \log Z_{\theta,x} \\
&= -\mathbb{E}_{x\sim p(x)} \frac{1}{Z_{\theta,x}}\ \nabla_\theta\, Z_{\theta,x} \\
&= -\mathbb{E}_{x\sim p(x)} \frac{1}{Z_{\theta,x}}\ \int (-\nabla_\theta\, H)\ \exp(-H)\ \mathrm{d}z \\
&= \mathbb{E}_{x\sim p(x)}\, \langle \nabla_\theta\, H \rangle
\end{aligned}
$$

Then with some effort of computation, we get

$$\begin{aligned}
A = \nabla_\theta^2 J(\theta,\lambda,\gamma) &= \nabla_\theta \, \mathbb{E}_{x\sim p(x)} \left[\frac{1}{Z_{\theta,x}} \int \nabla_\theta H \;\, \exp(-H) \; \mathrm{d}z\right] \\
&= \mathbb{E}_{x\sim p(x)} \left[\left\langle \nabla_\theta^2 H \right\rangle + \langle \nabla_\theta H\rangle \langle \nabla_\theta H\rangle^\top - \left\langle \nabla_\theta H \;\, \nabla_\theta^\top H\right\rangle\right]; \\
b_\lambda = -\frac{\partial}{\partial\lambda} \nabla_\theta J &= -\frac{\partial}{\partial\lambda}\mathbb{E}_{x\sim p(x)} \left[\frac{1}{Z_{\theta,x}} \int \nabla_\theta H \;\, \exp(-H) \; \mathrm{d}z\right] \\
&= -\mathbb{E}_{x\sim p(x)} \left[\left\langle \frac{\partial \nabla_\theta H}{\partial\lambda}\right\rangle - \left\langle \frac{\partial H}{\partial\lambda} \;\, \nabla_\theta H\right\rangle + \left\langle \frac{\partial H}{\partial\lambda}\right\rangle \langle \nabla_\theta H\rangle\right]; \\
b_\gamma = -\frac{\partial}{\partial\gamma} \nabla_\theta J &= -\frac{\partial}{\partial\gamma}\mathbb{E}_{x\sim p(x)} \left[\frac{1}{Z_{\theta,x}} \int \nabla_\theta H \;\, \exp(-H) \; \mathrm{d}z\right] \\
&= -\mathbb{E}_{x\sim p(x)} \left[\left\langle \frac{\partial \nabla_\theta H}{\partial\gamma}\right\rangle - \left\langle \frac{\partial H}{\partial\gamma} \;\, \nabla_\theta H\right\rangle + \left\langle \frac{\partial H}{\partial\gamma}\right\rangle \langle \nabla_\theta H\rangle\right].
\end{aligned}$$

According to the quasi-static constraints (2.21), we have

$$A\dot{\theta} - \dot{\lambda} b_\lambda - \dot{\gamma} b_\gamma = 0,$$

that implies

$$\dot{\theta} = A^{-1} b_\lambda \; \dot{\lambda} + A^{-1} b_\gamma \; \dot{\gamma} = \theta_\lambda \dot{\lambda} + \theta_\gamma \dot{\gamma}. \tag{2.38}$$

2.6.4. Computation of Iso-classification constraint

We start with computing the gradient of classification loss, clear that

$$C = \mathbb{E}_{x\sim p(x)} \left[-\int \; \mathrm{d}z \; e(z|x) \log c(y|z)\right] = -\mathbb{E}_{x\sim p(x)} \left\langle \ell \right\rangle,$$

where $\ell = \log c_\theta(y_x|z)$ is the logarithm of the classification loss, then

$$\begin{aligned}
\nabla_\theta\, C &= -\,\nabla_\theta\, \mathbb{E}_{x\sim p(x)}\left[\frac{1}{Z_{\theta,x}}\int \ell\ \exp(-H)\ \mathrm{d}z\right] \\
&= -\mathbb{E}_{x\sim p(x)}\left[\langle\nabla_\theta\ \ell\rangle + \langle\nabla_\theta\, H\rangle\,\langle\ell\rangle - \langle\ell\ \nabla_\theta\, H\rangle\right]; \\
\frac{\partial}{\partial\lambda}C &= -\frac{\partial}{\partial\lambda}\mathbb{E}_{x\sim p(x)}\left[\frac{1}{Z_{\theta,x}}\int \ell\ \exp(-H)\ \mathrm{d}z\right] \\
&= -\mathbb{E}_{x\sim p(x)}\left[-\frac{1}{Z_{\theta,x}^2}\left(\int -\frac{\partial H}{\partial\lambda}\ \exp(-H)\ \mathrm{d}z\right)\left(\int \ell\ \exp(-H)\ \mathrm{d}z\right) - \frac{1}{Z_{\theta,x}}\int \ell\ \frac{\partial H}{\partial\lambda}\ \exp(-H)\ \mathrm{d}z\right] \\
&= -\mathbb{E}_{x\sim p(x)}\left[\left\langle\frac{\partial H}{\partial\lambda}\right\rangle\langle\ell\rangle - \left\langle\ell\ \frac{\partial H}{\partial\lambda}\right\rangle\right];
\end{aligned}$$

The iso-classification loss constrains together with quasi-static constrains imply that:

$$\begin{aligned}
0 &\equiv \frac{\mathrm{d}}{\mathrm{d}t}C \\
&= \dot\theta^\top\ \nabla_\theta\, C + \dot\lambda\frac{\partial C}{\partial\lambda} + \dot\gamma\frac{\partial C}{\partial\gamma} \\
&= \dot\lambda\left(\theta_\lambda^\top\ \nabla_\theta\, C + \frac{\partial C}{\partial\lambda}\right) + \dot\gamma\left(\theta_\gamma^\top\ \nabla_\theta\, C + \frac{\partial C}{\partial\gamma}\right) \\
&= C_\lambda\dot\lambda + C_\gamma\dot\gamma,
\end{aligned}$$

where the third equation is followed by the equilibrium dynamics (2.22) for parameters θ. So far we developed the constrained dynamics for iso-classification process:

$$\begin{aligned}
0 &= C_\lambda\dot\lambda + C_\gamma\dot\gamma \\
\dot\theta &= \theta_\lambda\dot\lambda + \theta_\gamma\dot\gamma.
\end{aligned} \tag{2.39}$$

2.6.5. Iso-classification equations for changing data distribution

In this section we analyze the dynamics for iso-classification loss process when the data distribution evolves with time. $\frac{\partial p(x)}{\partial t}$ will lead to additional terms that represent the partial derivatives with respect to t on both the quasi-static and iso-classification constrains. More precisely, the new terms are

$$\begin{aligned} b_t &= -\frac{\partial}{\partial t} \nabla_\theta J = -\int \frac{\partial p(x)}{\partial t} \langle \nabla_\theta H \rangle \, \mathrm{d}x; \\ \frac{\partial}{\partial t} C &= -\int \frac{\partial p(x)}{\partial t} \langle \ell \rangle \, \mathrm{d}x, \end{aligned}$$

then the quasi-static and iso-classification constraints are ready to be modified as

$$\begin{aligned} 0 \equiv \frac{\mathrm{d}}{\mathrm{d}t} \nabla_\theta J(\theta, \lambda, \gamma) &\iff 0 = \nabla_\theta^2 F \; \dot{\theta} + \dot{\lambda} \; \frac{\partial \nabla_\theta F}{\partial \lambda} + \dot{\gamma} \; \frac{\partial \nabla_\theta F}{\partial \gamma} + \frac{\partial \nabla_\theta F}{\partial t} \\ &\iff \dot{\theta} = \dot{\lambda} \; A^{-1} \; b_\lambda + \dot{\gamma} \; A^{-1} \; b_\gamma + A^{-1} \; b_t \\ &\iff \dot{\theta} = \dot{\lambda} \; \theta_\lambda + \dot{\gamma} \; \theta_\gamma + \theta_t; \\ 0 \equiv \frac{\mathrm{d}}{\mathrm{d}t} C &\iff 0 = \dot{\theta}^\top \; \nabla_\theta C + \dot{\lambda} \frac{\partial C}{\partial \lambda} + \dot{\gamma} \frac{\partial C}{\partial \gamma} + \frac{\partial C}{\partial t} \\ &\iff 0 = \dot{\lambda} \left(\theta_\lambda^\top \; \nabla_\theta C + \frac{\partial C}{\partial \lambda} \right) + \dot{\gamma} \left(\theta_\gamma^\top \; \nabla_\theta C + \frac{\partial C}{\partial \gamma} \right) + \left(\theta_t^\top \; \nabla_\theta C + \frac{\partial C}{\partial t} \right) \\ &\iff 0 = \dot{\lambda} \; C_\lambda + \dot{\gamma} \; C_\gamma + C_t, \end{aligned}$$

where A, b_λ, b_γ, C_λ and C_γ where C_λ and C_γ are as given in lemma 6 and (2.26) with the only change being that the outer expectation is taken with respect to $x \sim p(x, t)$. The new terms that depends on time t are

$$C_t = -\int \frac{\partial p(x,t)}{\partial t} \langle \ell \rangle \, \mathrm{d}x - \mathbb{E}_{x \sim p(x,t)} \; \left[\left\langle \theta_t^\top \nabla_\theta H \right\rangle \langle \ell \rangle - \left\langle \theta_t^\top \nabla_\theta H \; \ell \right\rangle + \left\langle \theta_t^\top \nabla_\theta \ell \right\rangle \right] \tag{2.40}$$

with $\ell = \log c_\theta(y_{x_t} | z)$. We can combine modified quasi-static and iso-classification constraints to get

$$\begin{aligned} \dot{\theta} &= \left(\theta_\lambda - \frac{C_\lambda}{C_\gamma} \; \theta_\gamma \right) \dot{\lambda} + \left(\theta_t - \frac{C_t}{C_\gamma} \theta_\gamma \right) \\ &=: \hat{\theta}_\lambda \dot{\lambda} + \hat{\theta}_t \end{aligned}. \tag{2.41}$$

This indicates that $\theta = \theta(\lambda, t)$ is a surface parameterized by λ and t, equipped with a basis of tangent plane $(\hat{\theta}_\lambda, \hat{\theta}_t)$.

2.6.6. Optimally transporting the data distribution

We first give a brief description of the theory of optimal transportation. The optimal transport map between the source task and the target task will be used to define a dynamical process for the task. We only compute the transport for the inputs x between the source and target distributions and use a heuristic to obtain the transport for the labels y. This choice is made only to simplify the exposition; it is straightforward to handle the case of transport on the joint distribution $p(x, y)$.

If i.i.d samples from the source task are denoted by $\left\{x_1^s, \ldots, x_{n_s}^s\right\}$ and those of the target distribution are $\left\{x_1^t, \ldots, x_{n_t}^t\right\}$ the empirical source and target distributions can be written as

$$p^s(x) = \frac{1}{n_s} \sum_{i=1}^{n_s} \delta_{x - x_i^s}, \text{and } p^t(x) = \frac{1}{n_t} \sum_{i=1}^{n_t} \delta_{x - x_i^t}$$

respectively; here $\delta_{x-x'}$ is a Dirac delta distribution at x'. Since the empirical data distribution is a sum of a finite number of Dirac measures, this is a discrete optimal transport problem and easy to solve. We can use the Kantorovich relaxation to denote by $\mathcal{B}$ the set of probabilistic couplings between the two distributions:

$$\mathcal{B} = \left\{\Gamma \in \mathbb{R}_+^{n_s \times n_t} : \ \Gamma \mathbf{1}_{n_s} = p, \Gamma^\top \mathbf{1}_{n_t} = q\right\} \tag{2.42}$$

where $\mathbf{1}_n$ is an n-dimensional vector of ones. The Kantorovich formulation solves for

$$\Gamma^* = \underset{\Gamma \in \mathcal{B}}{\text{argmin}} \sum_{i=1}^{n_s} \sum_{t=1}^{n_t} \Gamma_{ij} \ \kappa_{ij} \tag{2.43}$$

where $\kappa \in \mathbb{R}_+^{n_s \times n_t}$ is a cost function that models transporting the datum x_i^s to x_j^t. This is the metric of the underlying data domain and one may choose any reasonable metric for $\kappa = \|x_i^s - x_j^t\|_2^2$. The problem in (2.43) is a convex optimization problem and can be solved easily; in practice we use the Sinkhorn's algorithm (Cuturi, 2013) which adds an entropic regularizer $-h(\Gamma) = \sum_{ij} \Gamma_{ij} \log \Gamma_{ij}$ to the objective in (2.43).

2.6.7. Changing the data distribution

Given the optimal probabilistic coupling Γ^* between the source and the target data distributions, we can interpolate between them at any $t \in [0, 1]$ by following the geodesics of the Wasserstein metric

$$p(x,t) = \operatorname*{argmin}_{p} \ (1-t)W_2^2(p^s, p) + tW_2^2(p, p^t).$$

For discrete optimal transport problems, as shown in Villani (2008), the interpolated distribution p_t for the metric $\kappa_{ij} = \|x_i^2 - x_j^t\|_2^2$ is given by

$$p(x,t) = \sum_{i=1}^{n_s} \sum_{j=1}^{n_t} \Gamma_{ij}^* \ \delta_{x-(1-t)x_i^s - tx_j^t}. \tag{2.44}$$

Observe that the interpolated data distribution equals the source and target distribution at $t = 0$ and $t = 1$ respectively and it consists of linear interpolations of the data in between.

Remark 12 (Interpolating the labels). The interpolation in (2.44) gives the marginal on the input space interpolated between the source and target tasks. To evaluate the functionals in Sec. 2.3 for the classification setting, we would also like to interpolate the labels. We do so by setting the true label of the interpolated datum $x = (1-t)x_i^s + tx_j^t$ to be linear interpolation between the source label and the target label.

$$y(x,t) = (1-t)\delta_{y-y_{x_i^s}} + t\delta_{y-y_{x_j^t}}$$

for all i, j. Notice that the interpolated distribution $p(x,t)$ is a sum of Dirac delta distributions weighted by the optimal coupling. We therefore only need to evaluate the labels at all the interpolated data.

Remark 13 (Linear interpolation of data). Our formulation of optimal transportation leads to a linear interpolation of the data in (2.29). This may not work well for image-based data where the square metric $\kappa_{ij} = \|x_i^s - x - k^t\|_2^2$ may not be the appropriate metric. We note that this interpolation of data is an artifact of our choice of κ_{ij}, other choices for the metric also fit into the formulation and should be viable alternatives if they result in efficient computation.

2.6.8. Details of the experimental setup for CIFAR transferring

At moment t, parameters λ, γ determine our objective functions. We compute iso-classification loss transfer process by first setting initial states: $(\lambda = 4, \gamma = 100)$. We train on source dataset for 300 epochs with Adam and a learning rate of 1E-3 that drops by a factor of 10 after every 120 epochs to obtain the initial state. We change λ, γ with respect to time t and then apply the equilibration learning rate schedule of Fig. 2.4a to achieve the transition between equilibrium states. We compute the partial derivatives $\frac{\partial C}{\partial t}$, $\frac{\partial C}{\partial \lambda}$ and $\frac{\partial C}{\partial \gamma}$ by using finite difference. At each time t, solving (2.37) with the partial derivatives leads to the solution for $\dot{\gamma}$, where $\dot{\lambda}$ is a constant. In our experiment we set $\dot{\lambda} = -1.5$.

2.6.9. Transfer learning with fine-tuning

A popular machine learning strategy is the transfer of a model learned on a source task to a target task. Examples include the re-use of neural network weights. In this section, we consider using the model from the source task to construct a prior, which is fine-tuned using target task data. We give a PAC-Bayes target task risk bound in this setting.

For target task we learn a posterior given a prior and training data. The quality of the prior affects the learner's performance. We proposes using source task to learn a 'hyperposterior'. Such a hyperposterior may focus the learner on a representation shared across source and target domain. More precisely, solving the Lagrangian in eq (2.11) gives the hyperposterior

$$p(\theta|\mathbf{D}_{\mathbf{s}}(\theta)) \propto m(\theta)e^{-[\hat{\mathbf{R}}_{\mathbf{s}}(\theta)(\theta)+\lambda\hat{\mathbf{D}}_{\mathbf{s}}(\theta)(\theta)+\gamma\hat{\mathbf{C}}_{\mathbf{s}}(\theta)(\theta)]/\sigma}, \tag{2.45}$$

we denote $\mathbf{D}_{\mathbf{s}}(\theta)$ and $\mathbf{D}_{\mathbf{t}}(\theta)$ as the source and target dataset respectively. Additionally, given the network parameter θ, $\hat{\mathbf{R}}_{\mathbf{s}}(\theta)(\theta)$, $\hat{\mathbf{D}}_{\mathbf{s}}(\theta)(\theta)$ and $\hat{\mathbf{C}}_{\mathbf{s}}(\theta)(\theta)$ are empirical functional on source dataset. Let $\mathcal{Z}$ be the partition function, the normalization constant for the hyperposterior

$$\mathcal{Z} := \int m(\theta)e^{-[\hat{\mathbf{R}}_{\mathbf{s}}(\theta)(\theta)+\lambda\hat{\mathbf{D}}_{\mathbf{s}}(\theta)(\theta)+\gamma\hat{\mathbf{C}}_{\mathbf{s}}(\theta)(\theta)]/\sigma}\mathrm{d}\theta. \tag{2.46}$$

Suppose $\mathbf{D}_{\mathbf{s}}(\theta)$ and $\mathbf{D}_{\mathbf{t}}(\theta)$ are consisted of $m_{\mathcal{S}}$ and $m_{\mathcal{T}}$ samples respectively. In case of the classification loss function is always bounded by 1. For $\theta \in \Theta$, let $\mathbf{C}_{\mathbf{t}}(\theta)(\theta)$ denote expected classification performance of model specified by θ on target domain.

We will consider the case that classification loss function is between 0 and 1, the general case follows by rescaling the loss function.

For positive ϵ, the Chernoff bound gives

$$P(|\mathbf{C_t}(\theta)(\theta) - \hat{\mathbf{C}}_\mathbf{t}(\theta)(\theta)| > \epsilon) \leq 2e^{-2m_{\mathcal{T}}\epsilon^2}. \tag{2.47}$$

According to Lemma 17 in "PAC bayesian Model Averaging", we have

$$\mathbb{E}_{\mathbf{D_t}(\theta)} e^{(2m_{\mathcal{T}}-1)|\mathbf{C_t}(\theta)(\theta) - \hat{\mathbf{C}}_\mathbf{t}(\theta)(\theta)|^2} \leq 4m_{\mathcal{T}}.$$

Therefore

$$\mathbb{E}_{\mathbf{D_t}(\theta)} \mathbb{E}_{\theta|\mathbf{D_s}(\theta)} e^{(2m_{\mathcal{T}}-1)|\mathbf{C_t}(\theta)(\theta) - \hat{\mathbf{C}}_\mathbf{t}(\theta)(\theta)|^2} \leq 4m_{\mathcal{T}}. \tag{2.48}$$

By Markov's inequality, for $1 > \delta > 0$, with prbability at least $1 - \delta$,

$$\mathbb{E}_{\theta|\mathbf{D_s}(\theta)} e^{(2m_{\mathcal{T}}-1)|\mathbf{C_t}(\theta)(\theta) - \hat{\mathbf{C}}_\mathbf{t}(\theta)(\theta)|^2} \leq 4m_{\mathcal{T}}/\delta. \tag{2.49}$$

We now consider selecting a posterior distribution (or density) $Q(\theta)$ on Θ, Jensen's inequality implies

$$\begin{aligned}\mathbb{E}_Q\left[(2m_{\mathcal{T}}-1)|\mathbf{C_t}(\theta)(\theta) - \hat{\mathbf{C}}_\mathbf{t}(\theta)(\theta)|^2 - \ln\frac{Q(\theta)}{p(\theta|\mathbf{D_s}(\theta))}\right] &\leq \ln \mathbb{E}_Q\left[\frac{p(\theta|\mathbf{D_s}(\theta))}{Q(\theta)} e^{(2m_{\mathcal{T}}-1)|\mathbf{C_t}(\theta)(\theta) - \hat{\mathbf{C}}_\mathbf{t}(\theta)(\theta)|^2}\right] \\ &\leq \ln(4m_{\mathcal{T}}/\delta).\end{aligned}$$

Finally, with probability at least $1 - \delta$, we have

$$\mathbb{E}_{\theta\sim Q}|\mathbf{C_t}(\theta)(\theta) - \hat{\mathbf{C}}_\mathbf{t}(\theta)(\theta)|^2 \leq \frac{D_{KL}(Q(\theta)\|p(\theta|\mathbf{D_s}(\theta))) + \ln\frac{4m_{\mathcal{T}}}{\delta}}{2m_{\mathcal{T}}-1}.$$

The dominant term in right hand side of the above inequality is the KL divergence between posterior Q and the prior pretrained on source task. With some effort of computation, we have

$$D_{KL}(Q(\theta)\|p(\theta|\mathbf{D_s}(\theta)) = \frac{1}{\sigma}\mathbb{E}_{\theta\sim Q}[\hat{\mathbf{R}}_\mathbf{s}(\theta)(\theta) + \lambda\hat{\mathbf{D}}_\mathbf{s}(\theta)(\theta) + \gamma\hat{\mathbf{C}}_\mathbf{s}(\theta)(\theta)] + D_{KL}(Q(\theta)\|m(\theta)) + \log\mathcal{Z}$$

Furthermore, we see that minimizing the train error $\mathbb{E}_{\theta\sim Q}\hat{\mathbf{C}}_{\mathbf{t}}(\theta)(\theta)$ together with $D_{KL}(Q(\theta)\|p(\theta|\mathbf{D}_{\mathbf{s}}(\theta)))$ can be interpreted as minimizing an upper-bound on the test error $\mathbb{E}_{\theta\sim Q}\mathbf{C}_{\mathbf{t}}(\theta)(\theta)$ of the model, rather than directly minimizing the train error. This observation inspires us a new fine tune framework, by introducing a hyper parameter β

$$\min_{Q}\mathbb{E}_{\theta\sim Q}\left[\hat{\mathbf{C}}_{\mathbf{t}}(\theta)(\theta)+\frac{\beta}{\sigma}(\hat{\mathbf{R}}_{\mathbf{s}}(\theta)(\theta)+\lambda\hat{\mathbf{D}}_{\mathbf{s}}(\theta)(\theta)+\gamma\hat{\mathbf{C}}_{\mathbf{s}}(\theta)(\theta))\right]+\beta D_{KL}(Q(\theta)\|m(\theta))$$

This is in accordance with the intuition developed earlier, that minimizing

$$\mathbb{E}_{\theta\sim Q}\left[\frac{\beta}{\sigma}(\hat{\mathbf{R}}_{\mathbf{s}}(\theta)(\theta)+\lambda\hat{\mathbf{D}}_{\mathbf{s}}(\theta)(\theta)+\gamma\hat{\mathbf{C}}_{\mathbf{s}}(\theta)(\theta))\right]$$

forces the model to capture the information from source domain that would be potentially useful in target domain, and $D_{KL}(Q(\theta)\|m(\theta))$ reduce the model complexity.

CHAPTER 3

AN INFORMATION GEOMETRIC DISTANCE ON THE SPACE OF TASKS

A representation that adheres to the free energy principle (2.7) preserves the additional information and models the data-generating process aligned with the pre-training source task, denoted as $p(x, y)$. To better transfer such a representation to adapt to a new target task $p^{\text{new}}(x, y)$, it requires us to navigate the tasks from $p(x, y)$ to $p^{\text{new}}(x, y)$ properly in the space of the tasks. In Sec. 2.4, we transport the task from $p(x, y)$ to $p^{\text{new}}(x, y)$ using the mixture interpolation,

$$p^t = (1-t)p + tp^{\text{new}},$$

for $0 \leq t \leq 1$. However, the mixture interpolation can not represent the optimal way to move the tasks. This problem serves as our motivation for this chapter. By leveraging optimal transportation (OT), we establish a sequence of interpolated tasks that evolves from $p(x, y)$ to $p^{\text{new}}(x, y)$. The representation is then updated to align with the evolving data distribution. We refer to this process as *optimal coupled transfer*. Optimal coupled transfer facilitates model transfer, surpassing the direct fine-tuning approach on the target task. It enables the pre-trained model to traverse the shortest path in the space of tasks, thereby adapting to the new task efficiently.

From an information geometric perspective, the length of this shortest path connecting two tasks gives rise to a unique definition of the distance between them. Consequently, we address a longstanding open question: how to define the distance between tasks theoretically soundly. We provide experimental evidence to support our viewpoints. Through minor modifications in the code, we update models to adapt to the sequential interpolated tasks. The results outperform the fine-tuning approach.

We are interested in the supervised learning problem in this chapter. Consider a source dataset $D_s = \left\{(x_s^i, y_s^i)\right\}_{i=1}^{N_s}$ and a target dataset $D_t = \left\{(x_t^i, y_t^i)\right\}_{i=1}^{N_t}$ where $x_s^i, x_t^i \in X$ denote input data and $y_s^i, y_t^i \in Y$ denote ground-truth annotations. Training a parameterized classifier, say a deep network with weights $w \in \mathbb{R}^p$, on the source task, involves minimizing the cross-entropy loss $\ell_s(w) = -\frac{1}{N_s} \sum_{i=1}^{N_s} \log p_w(y_s^i | x_s^i)$ using stochastic gradient descent (SGD):

$$w(\tau + \mathrm{d}\tau) = w(\tau) - \hat{\nabla} \ell_s(w(\tau))\ \mathrm{d}\tau;\ w(0) = w_s; \tag{3.1}$$

The notation $\widehat{\nabla} \ell_s(w)$ indicates a stochastic estimate of the gradient using a mini-batch of data. The parameter $\mathrm{d}\tau$ is the learning rate. Let us define the distribution $\hat{p}_s(x, y) = \frac{1}{N_s} \sum_{i=1}^{N_s} \delta_{x_s^i}(x) \delta_{y_s^i}(y)$ and its input-marginal $\hat{p}_s(x) = \frac{1}{N_s} \sum_{i=1}^{N_s} \delta_{x_s^i}(x)$; distributions $\hat{p}_t(x, y), \hat{p}_t(x)$ are defined analogously.

3.1. An Overview of Measuring the Distances between Tasks

A part of the success of Deep Learning stems from the fact that deep networks learn features that are discriminative yet flexible. There is a prevailing belief in the research community that deep learning tasks exhibit inherent relationships that signify similarities or dissimilarities in underlying patterns. Therefore, Models pre-trained on a particular task could be easily adapted to perform well on other tasks. The transfer learning literature forms an umbrella for such adaptation techniques, and it works well, see for instance Mahajan et al. (2018); Dhillon et al. (2020); Kolesnikov et al. (2019); Joulin et al. (2016); Song et al. (2020) for image classification or Devlin et al. (2018) for language modeling, to name a few large-scale studies. There are also situations when transfer learning does not work well, e.g., a pre-trained model on ImageNet is a poor representation to transfer to MRI data (Merkow et al., 2017).

It stands to reason that if source and target tasks are "close" to each other then we should expect transfer learning to work well. It may be difficult to transfer across tasks that are "far away". Researchers should not be satisfied with the ambiguous and empirical descriptions such as: "Task A is close with Task B, but far away with Task C". However, the vast diversity in deep learning configurations, including dataset composition, network architectures, optimization methods, and transfer learning mechanisms, presents a major challenge in studying the relationship among typical learning tasks. This diversity complicates the building of a universal framework that quantifies task relationships. This section introduces the previous research works on defining the similarity between tasks.

3.1.1. Discrepancy measures on the input space

The natural language processing literature presents several methods to directly compute the similarity between task input data (Mikolov et al., 2013; Pennington et al., 2014). These methods think of the tasks as data distributions and then compute *Wasserstein distance*, Maximum Mean Discrepancy (MMD), Hellinger distance, or other f-divergences to measure distances between probability distributions. In this section, we only introduce the method for computing Wasserstein distance.

We focus on the marginals on the input data $\hat{p}_s(x)$ and $\hat{p}_t(x)$ for the source and target tasks, respectively. We compute the Wasserstein distance between the source marginal and the target marginal and will use tools from optimal transportation (OT) for this purpose; see Santambrogio (2015); Peyré and Cuturi (2019); Fatras et al. (2020) for an elaborate treatment.

OT for continuous measures Let $\Pi(p_s, p_t)$ be the set of joint distributions (also known as couplings or transport plans) with the first marginal equal to $p_s(x)$ and the second marginal $p_t(x)$. The Kantorovich relaxation of OT solves for

$$\inf_{\gamma \in \Pi(p_s, p_t)} \int c(x, x') \, \mathrm{d}\gamma(x, x')$$

to compute the best coupling $\gamma^* \in \Pi$. The cost $c(x, x') \in \mathbb{R}_+$ is called the ground metric. It gives the cost of transporting unit mass from x to x'. The popular squared-Wasserstein metric $W_2^2(p_s, p_t)$ uses $c(x, x') = \|x - x'\|_2^2$. Given the optimal coupling γ^*, we can compute the trajectory that transports probability mass using displacement interpolation (McCann, 1997). For example, for the Wasserstein metric, γ^* is a constant-speed geodesic, i.e., if p_τ is the distribution at an intermediate time instant $\tau \in [0, 1]$ then its distance from p_s is proportional to τ

$$W_2(p_s, p_\tau) = \tau W_2(p_s, p_t).$$

OT for discrete measures In case of the discrete measures $\hat{p}_s(x)$ and $\hat{p}_t(x)$, the set of transport plans in this case is $\Pi(\hat{p}_s, \hat{p}_t) = \left\{ \Gamma \in \mathbb{R}_+^{N_s \times N_t} : \; \Gamma \mathbf{1}_{N_s} = \hat{p}_s, \Gamma^\top \mathbf{1}_{N_t} = \hat{p}_t \right\}$ and the optimal coupling is given by

$$\Gamma^* = \underset{\Gamma \in \Pi(\hat{p}_s, \hat{p}_t)}{\operatorname{argmin}} \; \{ \langle \Gamma, C \rangle - \epsilon H(\Gamma) \} \, ; \tag{3.2}$$

here C_{ij} is a matrix that defines the ground metric in OT. For instance, $C_{ij} = \|x_i - x'_j\|_2^2$ for the Wasserstein metric. The first term above measures the total cost $\sum_{ij} \Gamma_{ij} C_{ij}$ incurred for the transport. The second term is an entropic penalty $H(\Gamma) = -\sum_{ij} \Gamma_{ij} \log \Gamma_{ij}$ popularized by Cuturi (2013) that accelerates the solution of the OT problem. McCann's interpolation for the discrete case with $C_{ij} = \|x_s^i - x_t^j\|_2^2$ can be written explicitly as a sum of Dirac-delta distributions supported at interpolated inputs $x_\tau^{ij} = (1-\tau) x_s^i + \tau x_t^j$

$$\hat{p}_\tau(x) = \sum_{i=1}^{N_s} \sum_{j=1}^{N_t} \Gamma_{ij}^* \; \delta_{(1-\tau)x_s^i + \tau x_t^j}(x). \tag{3.3}$$

We can also create pseudo labels for samples from p_τ by linear interpolation of the one-hot encoding of their respective labels to get

$$\hat{p}_\tau(x, y) = \sum_{i=1}^{N_s} \sum_{j=1}^{N_t} \Gamma_{ij}^* \; \delta_{(1-\tau)x_s^i + \tau x_t^j}(x) \; \delta_{(1-\tau)y_s^i + \tau y_t^j}(y). \tag{3.4}$$

Remark 14 (Measuring distance between learning tasks is different than measuring distances between the respective data distributions). The above discrepancy concepts can only measure distances between data distributions. They do not consider the hypothesis class used to transfer across the two distributions and therefore do not reflect the true difficulty of transfer. The experiment in Fig. 3.7 demonstrates this. This point, in fact, is the central motivation of this chapter.

3.1.2. **Task2Vec**

Task2Vec (Achille et al., 2019a) provide vectorial representations of visual classification tasks, which can be used to reason about the nature of those tasks and their relations. The definition of task2vec is grounded in the Fisher information matrix (FIM). Not all combinations of the network weights are equally important in predicting the task variable: the importance, or "informative content," of weight for the task can be quantified by considering a small perturbation $w + \mathrm{d}w$ of the weights and measuring the average Kullbach-Leibler (KL) divergence between the original output distribution $p_w(y \,|\, x)$ and the perturbed one $p_{w+\mathrm{d}w}(y \,|\, x)$. To second-order approximation, this is

$$\mathbb{E}_{x\sim\hat{p}}\mathrm{KL}(p_{w+\mathrm{d}w}(y \,|\, x),\ p_w(y \,|\, x)) = \mathrm{d}w^T G \mathrm{d}w + o(\|\mathrm{d}w\|^2),$$

$\hat{p}$ is an empirical distribution of the input data, and G is the Fisher information matrix (FIM):

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

FIM G is the expected covariance of the scores (gradients of the log-likelihood) with respect to the model parameters. FIM provides a measure of the information a particular parameter (weight or feature) contains about the joint distribution $\hat{p}(x) \cdot p_w(y \,|\, x)$: If the classification performance for a given task does not depend strongly on a parameter, the corresponding entry in the FIM will be small.

While the network activations capture the information in the input image which is needed to infer the image label, the FIM indicates the set of feature maps that are more informative for solving the current task. Following this intuition, Task2Vec uses the FIM to represent the task itself. However, the FIMs computed on different networks are not directly comparable. To address this, Task2Vec uses 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.

Let $\mathrm{d}_{\cos}$ denote the cosine distance between two vectors. Let G_s and G_t denote the task embedding (i.e., the diagonal of the Fisher Information computed on the same probe network) for the source and target tasks, respectively. Task2Vec distances compute the cosine distance between normalized FIM embedding,

$$\mathrm{d}_{\cos}\left(\frac{G_s}{G_s + G_t},\ \frac{G_t}{G_s + G_t}\right), \tag{3.5}$$

and the division is element-wise.

Remark 15 (Task2Vec does not align with the fine-tuning difficulty). The FIM is also related to the (Kolmogorov) complexity of a task(Achille et al.). The norm of the embedding correlates with the complexity of the task, while the distance between embeddings captures semantic similarities between tasks. The main hurdle in Task2Vec and similar approaches is to design the architecture for computing FIM: a small model will indicate that tasks are far away. This is what we always claim: defining the distance between learning tasks needs to take the hypothesis class into consideration. In addition, our experimental results indicate that Task2Vec does not align well with the fine-tuning difficulty.

3.1.3. Other measurements on the distances between tasks

Taskonomy (Zamir et al., 2018) focuses on the network architectures consisting of the feature extractor and the linear classifier. It pre-trains a model on the source task, then freezes the feature extractor and re-trains a linear classifier to adapt the target task. The prediction performance on the target task measures the similarity between the source and target tasks.

There are also classical trivial measurements such as the number of epochs to reach a standard accuracy while transfer learning from the source to the target task, the length of trajectories in the weight space $\int_{w_s}^{w_t} |\mathrm{d}w|$, and the discrepancy measures on the representation space computed on a shared probe network.

3.2. An Information Geometric Distance Between the Tasks

We regard the tasks as the joint probability distributions $p(x, y)$ between the inputs and labels. Let $z = \{x, y\}$ denote the collection of the input and the output if we do not wish to distinguish inputs and labels. Information geometry is a well-established set of tools designed precisely for understanding the geometric properties of probability distributions. Consider a manifold $\mathcal{M} = \{p_w(z) : w \in \mathbb{R}^p\}$ of probability distributions parameterized by w. Information Geometry (Amari, 2016b) studies invariant geometrical structures on such manifolds. For two points $w, w' \in \mathcal{M}$, we can use the Kullback-Leibler (KL) divergence

$$\mathrm{KL}\left[p_w, p_{w'}\right] = \int \mathrm{d}z\ p_w(z) \log \frac{p_w(z)}{p_{w'}(z)},$$

to obtain a Riemannian structure on M. This allows the infinitesimal distance $\mathrm{d}s$ on the manifold to be written as

$$\mathrm{d}s^2 = 2\mathrm{KL}\left[p_w, p_{w+\mathrm{d}w}\right] = \sum_{i,j=1}^{p} g_{ij}\ \mathrm{d}w_i \mathrm{d}w_j \tag{3.6}$$

$$g_{ij}(w) = \int \mathrm{d}z\ p_w(z) \left(\partial_{w_i} \log p_w(z)\right) \left(\partial_{w_j} \log p_w(z)\right) \tag{3.7}$$

are elements of the Fisher Information Matrix (FIM) g. Weights w play the role of a coordinate system for computing the distance. The FIM is the Hessian of the KL-divergence; we may think of the FIM as quantifying the amount of information present in the model about the data it was trained on. The FIM is the unique metric on $\mathcal{M}$ (up to scaling) that is preserved under diffeomorphisms (Bauer et al., 2016), in particular under representation of the model.

Given a continuously differentiable curve $\{w(\tau)\}_{\tau\in[0,1]}$ on the manifold M, we can compute its length by integrating the infinitesimal distance $|\mathrm{d}s|$ along it. The shortest length curve between two points $w, w' \in \mathcal{M}$ induces a metric on $\mathcal{M}$ known as the Fisher-Rao distance (Rao, 1945)

$$d_{\mathrm{FR}}(w, w') = \min_{\substack{\{w(\tau)\}:\ w(0)=w \\ w(1)=w'}} \int_0^1 \sqrt{\langle \dot{w}(\tau), g(w(\tau)) \dot{w}(\tau) \rangle}\ \mathrm{d}\tau \tag{3.8}$$

The shortest paths on a Riemannian manifold are geodesics, i.e., they are locally "straight lines".

Computing the Fisher-Rao distance by integrating the KL-divergence Let us focus on the conditional distribution $p_w(y\,|\,x)$. For the factorization $p(x,y) = p(x)p(y\,|\,x)$ where only the latter is parametrized, the FIM in (3.7) is given by

$$g_{ij}(w) = \mathbb{E}_{x\sim p(x),\ y\sim p_w(y|x)} \left[\partial_{w_i} \log p_w(y|x)\ \partial_{w_j} \log p_w(y|x)\right]$$

here the input distribution $p(x)$ and the weights w will be chosen in the following sections. The

FIM is difficult to compute for large models, and approximations often work poorly (Kunstner et al., 2019). For our purposes, we only need to compute the infinitesimal distance $|\mathrm{d}s|$ in (3.6) and can thus rewrite (3.8) as

$$d_{\mathrm{FR}}(w, w') = \min_{\substack{\{w(\tau)\}:\ w(0)=w \\ w(1)=w'}} \int_0^1 \sqrt{2\mathbb{E}_{x\sim p(x)}\mathrm{KL}[p_{w(\tau)}(\cdot\,|\,x), p_{w(\tau+\mathrm{d}\tau)}(y\,|\,x)]}. \tag{3.9}$$

We next combine the development of measuring the length of curves (3.9) and optimal transportation for discrete measures (3.4). We transport the margin on the data and modify the model weights simultaneously. We call this method the *coupled transfer process* and the corresponding task distance as the *coupled transfer distance*. We also discuss techniques to efficiently implement the process and make it scalable to large deep networks.

3.2.1. **Uncoupled transfer distance**

We first discuss a simple transport mechanism instead of OT and discuss how to compute a transfer distance. For $\tau \in [0, 1]$, consider the mixture distribution

$$\hat{p}_\tau(x, y) = (1-\tau)\hat{p}_s(x, y) + \tau\hat{p}_t(x, y). \tag{3.10}$$

Samples from $\hat{p}_\tau$ can be drawn by sampling an input-output pair from $\hat{p}_s$ with probability $1-\tau$ and sampling it from $\hat{p}_t$ otherwise. At each time instant τ, the uncoupled transfer process updates the weights of the classifier using SGD to fit samples from $\hat{p}_\tau$

$$w(\tau + \mathrm{d}\tau) = w(\tau) - \hat{\nabla}\ell_\tau(w(\tau))\ \mathrm{d}\tau;\ w(0) = w_s. \tag{3.11}$$

Weights $w(\tau)$ are thus fitted to each task p_τ as τ goes from 0 to 1. In particular for $\tau = 1$, weights $w(1)$ are fitted to $\hat{p}_t$. As $\mathrm{d}\tau \to 0$, we obtain a continuous curve $\{w(\tau) : t \in [0, 1]\}$. Computing the length of this weight trajectory gives a transfer distance analogy to (3.9),

$$\int_0^1 \sqrt{\mathbb{E}_{x\sim\hat{p}_\tau} 2\mathrm{KL}\left[p_{w(\tau)}(\cdot\,|\,x), p_{w(\tau+\mathrm{d}\tau)}(\cdot\,|\,x)\right]}. \tag{3.12}$$

Remark 16 (Uncoupled transfer distance entails longer weight trajectories). For uncoupled transfer, although the task and weights are modified simultaneously, their changes are not synchronized. We, therefore, call this the uncoupled transfer distance. To elucidate, changes in the data using the mixture (3.10) may be unfavorable to the current weights $w(\tau)$ and may cause the model to struggle to track the distribution $\hat{p}_\tau$. This forces the weights to take a longer trajectory in prediction space, i.e., the analogy to be measured by the Fisher-Rao distance in (3.9). If changes in data were synchronized with the evolving weights, the weight trajectory would be necessarily shorter in prediction space because the KL-divergence in (3.9) is large when the conditional distribution changes quickly to track the evolving data. We therefore expect the task distance computed using the mixture distribution to be larger than the coupled transfer distance, which we will discuss next; our experiments in Sec. 3.4 corroborate this.

3.2.2. Modifying the task and classifier synchronously

Our coupled transfer distance that uses OT to modify the task and updates the weights synchronously (as shown in Fig. 3.1) to track the interpolated distribution is defined as follows.

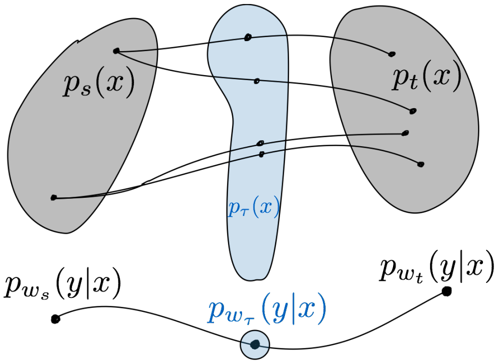


Figure 3.1: **Coupled transfer of the data and the conditional distribution**. We solve an optimization problem that transports the source data distribution $p_s(x)$ to the target distribution $p_t(x)$ as $\tau \to 1$ while simultaneously updating the model using samples from the interpolated distribution $p_\tau(x)$. This modifies the conditional distribution $p_{w_s}(y \mid x)$ on the source task to the corresponding distribution on the target task $p_{w_t}(y \mid x)$. The coupled transfer distance between source and target tasks is the length of the shortest such weight trajectory under the Fisher Information Metric.

Definition 17 (Coupled transfer distance). Given two learning tasks D_s and D_t and a w-parametrized classifier trained on D_s with weights w_s, the coupled transfer distance between the tasks is

$$\min_{\Gamma, w(\cdot)} \mathbb{E}_{x \sim \hat{p}_\tau} \int_0^1 \sqrt{2\mathrm{KL}\left[p_{w(\tau)}(\cdot \,|\, x), p_{w(\tau+\mathrm{d}\tau)}(\cdot \,|\, x)\right]} \tag{3.13}$$

where and couplings $\Gamma \in \Pi(\hat{p}_s(x), \hat{p}_t(x))$ and $w(\cdot)$ is a continuous curve which is the limit of

$$w(\tau + \mathrm{d}\tau) = w(\tau) - \hat{\nabla}\ell_\tau(w(\tau))\ \mathrm{d}\tau;\ w(0) = w_s.$$

as $\mathrm{d}\tau \to 0$. The interpolated distribution $\hat{p}_\tau(x, y)$ at time instant $\tau \in [0, 1]$ for a coupling Γ is given by (3.4) and the loss ℓ_τ is the cross-entropy loss of fitting data from this interpolated distribution.

In comparison of (3.12), we move the expectation
$E_{x \sim \hat{p}_\tau}$ outside the square root for simplifying the computation. The following remarks discuss the rationale and the properties of this definition.

Remark 18 (Coupled transfer distance is asymmetric). The length of the weight trajectory for transferring from $\hat{p}_s$ to $\hat{p}_t$ is different from the one that transfers from $\hat{p}_t$ to $\hat{p}_s$. This is a desirable property, e.g., it is easier to transfer from ImageNet to CIFAR-10 than in the opposite direction.

Remark 19 (Coupled transfer distance can be compared across different architectures). An important property of the task distance in (3.13) is that it is the Fisher-Rao distance, i.e., the shortest geodesic on the statistical manifold, of conditional distributions $p_{w(0)}(\cdot \,|\, x_s^i)$ and $p_{w(1)}(\cdot \,|\, x_t^i)$ with the coupling Γ determining the probability mass that is transported from x_s^i to x_t^j. Since the Fisher-Rao distance does not depend on the embedding dimension of the manifold M, the coupled transfer distance does not depend on the architecture of the classifier; it only depends upon the capacity to fit the conditional distribution $p_w(y \,|\, x)$. This is a very desirable property: given the tasks, our distance is comparable across different architectures. Let us note that the uncoupled transfer distance in Sec. 3.2.1 also shares this property, but the coupled transfer has the benefit of computing the shortest trajectory in information space; weight trajectories of uncoupled transfer may be larger; see Rem. 16.

3.2.3. **Computing the coupled transfer distance**

We first provide an informal description of how we compute the task distance. Each entry Γ_{ij} of the coupling matrix determines how much probability mass from x_s^i is transported to x_t^j. The interpolated distribution (3.4) allows us to draw samples from the task at an intermediate instant. For each coupling Γ, there exists a trajectory of weights $w(\cdot) := \{w(\tau) : \tau \in [0,1]\}$ that tracks the interpolated task. The algorithm treats Γ and the weight trajectory as the two variables and updates them alternately as follows. At the k^{th} iteration, given a weight trajectory $w^k(\cdot)$ and a coupling Γ^k, we set the entries of the ground metric C_{ij}^{k+1} to be the Fisher-Rao distance between distributions $p_{w(0)}(\cdot \,|\, x_s^i)$ and $p_{w(1)}(\cdot \,|\, x_t^i)$. An updated Γ^{k+1} is calculated using this ground metric to result in a new trajectory $w^{k+1}(\cdot)$ that tracks the new interpolated task distribution (3.4) for Γ^{k+1}.

More formally, given an initialization for the coupling matrix Γ^0 we perform the updates in (3.14). Computing the coupled transfer distance is a non-convex optimization problem and we therefore include a proximal term in (3.14) to keep the coupling matrix close to the one computed in the previous step Γ^k. This also indirectly keeps the weight trajectory $w^{k+1}(\cdot)$ close to the trajectory from the previous iteration. Proximal point iteration (Bauschke and Combettes, 2017) is insensitive to the step-size λ and it is therefore beneficial to employ it in these updates.

$$\Gamma^k = \underset{\Gamma \in \Pi}{\operatorname{argmin}} \left\{ \left\langle \Gamma, C^k \right\rangle - \epsilon H(\Gamma) + \lambda \|\Gamma - \Gamma^{k-1}\|_{\mathrm{F}}^2 \right\}, \tag{3.14}$$

$$C_{ij}^k = \int_0^1 \sqrt{2\mathrm{KL}\left[p_{w^k(\tau)}(\cdot \,|\, x_\tau^{ij}),\ p_{w^k(\tau+\mathrm{d}\tau)}(\cdot \,|\, x_\tau^{ij})\right]}, \tag{3.15}$$

$$w^k(\tau + \mathrm{d}\tau) = w^k(\tau) - \hat{\nabla}\ell_\tau(w^k(\tau))\,\mathrm{d}\tau, \quad w(0) = w_{s^*}, \tag{3.16}$$

$$\hat{p}_\tau(x,y) = \sum_{i=1}^{N_s} \sum_{j=1}^{N_t} \Gamma_{ij}^{k-1}\ \delta_{(1-\tau)x_s^i + \tau x_t^j}(x)\ \delta_{(1-\tau)y_s^i + \tau y_t^j}(y) \tag{3.17}$$

$$x_\tau^{ij},\ y_\tau^{ij} \sim \hat{p}_\tau(x,y).$$

3.2.4. Practical tricks for efficient computation

The optimization problem formulated in (3.14) is conceptually simple but computationally daunting. The main hurdle is to compute the ground metric C^k_{ij} for all $i \leq N_s, j \leq N_t$ pairs in a dense transport coupling Γ. The coupling matrix can be quite large, e.g., it has 10^8 entries for a relatively small dataset of $N_s = N_t = 10,000$. We therefore introduce the following techniques that allow us to scale to large problems.

Block-diagonal transport couplings Instead of optimizing Γ in (3.13) over the entire polytope $\Pi(\hat{p}_s, \hat{p}_t)$, we only consider block-diagonal couplings. Depending upon the source and target datasets, we use blocks of size up to 30×30. At each time instant $\tau \in [0, 1]$, we sample a block from the transport coupling. SGD in (3.16) updates weights using multiple samples from the interpolated task restricted to this block. The integrand for C^k_{ij} in (3.15) is also computed only on this mini-batch. Experiments in Sec. 3.4 show that the weight trajectory converges using this technique. We can compute the coupling transfer distance for source and target datasets of size up to $N_s = N_t = 19,200$. Other approaches for handling large-scale OT problems such as hierarchical methods (Lee et al., 2019) or greedy computation (Carlier et al., 2010) could also be used for our purpose but we chose this one for sake of simplicity.

Initializing the transport coupling The ground metric $C_{ij} = \|x^i_s - x^j_t\|^2_2$ is widely used in the OT literature. We are however interested in computing distances for image-classification datasets in this paper and such a pixel-wise distance is not a reasonable ground metric for visual data that have strong local/multi-scale correlations. We therefore set Γ^0 to be the block-diagonal approximation of the transport coupling for the ground metric $C_{ij} = \|\varphi(x^i_s) - \varphi(x^j_t)\|^2_2$ where φ is some feature extractor. The feature space is much more Euclidean-like than the input space and this gives us a good initialization in practice; similar ideas are employed in the metric learning literature (Snell et al., 2017; Hu et al., 2015; Qi et al., 2018). We use a ResNet-50 He et al. (2016b) pre-trained on ImageNet to initialize Γ^0 for all our experiments. To emphasize, *we use the feature extractor only for initializing the transport coupling* further updates are performed using (3.14). We have computed the coupling transfer distance for MNIST without this step and our results are similar.

Using mixup to interpolate source and target images The interpolating distribution (3.4) has a peculiar nature: sampled data $x_\tau^{ij} = (1-\tau)x_s^i + \tau x_t^j$ from this distribution are a convex combination of source and target data. This causes artifacts for natural images for τ away from 0 or 1; we diagnosed this as a large value of the training loss while executing (3.11). We therefore treat the coefficient of the convex combination in (3.4) as if it were a sample from a Beta-distribution $\text{Beta}(\tau, 1-\tau)$. This keeps the samples x_τ^{ij} similar to the source or the target task and avoids visual artifacts. This trick is inspired by Mixup regularization Zhang et al. (2017); we also use Mixup for labels y_τ^{ij}.

3.3. An Alternative Perspective using Rademacher Complexity

We have hitherto motivated the coupled transfer distance using ideas in information geometry. In this section, we study the weight trajectory under the lens of learning theory. We show that we can interpret it as the trajectory that minimizes the integral of the generalization gap as the the weights are adapted from the source to the target task. We consider binary classification tasks in this section. Rademacher complexity (Bartlett and Mendelson, 2001)

$$\mathcal{R}_N(r) = \mathbb{E}_{\hat{p}\sim p}\left[\mathbb{E}_\sigma\left[\sup_{w\in A(r)} \frac{1}{N}\sum_{i=1}^{N} \sigma^i \ell(w; x^i, y^i)\right]\right], \tag{3.18}$$

is the average over draws of the dataset $\hat{p} \sim p$ and iid random variables σ^i uniformly distributed over $\{-1, 1\}$ of the worst case average weighted loss $\sigma^i \ell(w; x^i, y^i)$ for w in the set $A(r)$. We assume here that $\left|\ell(w; x^i, y^i)\right| < M$ and $\ell(w; x, y)$ is Lipschitz continuous. Classical bounds bound the generalization gap of all hypotheses h in a hypothesis class $\mathcal{H}$ by $\mathcal{R}_{2N}(\mathcal{H}) + 2\sqrt{\frac{\log(1/\delta)}{N}}$ with probability at least $1-\delta$. We build upon this result to get the following theorem under the assumption that weights $w(\tau)$ predict well on the interpolated task $\hat{p}_\tau(x, y)$ at all times τ.

Theorem 20. Given a weight trajectory $\{w(\tau)\}_{\tau\in[0,1]}$ and a sequence $0 = \tau_0 \leq \tau_1 < \tau_2 < ... < \tau_K \leq 1$, for all $\epsilon > 2\sum_{k=1}^{K}(\tau_k - \tau_{k-1})\mathbb{E}_{x\sim p_\tau}|\Delta\ell(w(\tau_{k-1}))|$, the probability that

$$\frac{1}{K}\sum_{k=1}^{K}\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}}\left[\ell(\omega(\tau_k), x, y)\right] - \frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}} \ell(\omega(\tau_k), x, y)\right)$$

is greater than ϵ is upper bounded by

$$\exp\left\{-\frac{2K}{M^2}\left(\epsilon - 2\sum_{k=1}^{K}\Delta\tau_k \mathbb{E}_{x\sim p_{\tau_k}}\left[\sqrt{\langle \dot{w}(\tau_k), g(w(\tau_k))\dot{w}(\tau_k)\rangle}\right]\right)\right\}. \tag{3.19}$$

We have defined $\Delta\tau_k = \tau_k - \tau_{k-1}$ and $\Delta\ell(w(\tau)) = \ell(w(\tau + \mathrm{d}\tau); x, y_\tau(x)) - \ell(w(\tau); x, y_\tau(x))$.

Sec. 3.6.6 gives the proof. As $\Delta\tau_k \to 0$

$$\sum_{k=1}^{K}\Delta\tau_k \mathbb{E}_{x\sim p_{\tau_k}}\left[\sqrt{\langle \dot{w}(\tau_k), g(w(\tau_k))\dot{w}(\tau_k)\rangle}\right] \to \int_0^1 \mathbb{E}_{x\sim \hat{p}_\tau}\left[\sqrt{\langle \dot{w}, g(w)\dot{w}\rangle}\right] \mathrm{d}\tau$$

which is the length of the trajectory on the statistical manifold with inputs drawn from the interpolated distribution at each instant.

We can thus think of the coupled transfer distance as the length of the trajectory on the statistical manifold that starts at the given model w_s on the source task and ends with the model $w(1)$ fitted to the target task, as the task is simultaneously interpolated using an optimal transport whose ground metric between samples x_s^i and x_t^j is $C_{ij} = \int_0^1 \sqrt{2\mathrm{KL}\left[p_{w(\tau)}(\cdot|x_\tau^{ij}), p_{w(\tau+\mathrm{d}\tau)}(\cdot|x_\tau^{ij})\right]}$ which is the length of the trajectory under the FIM. This result is a crisp theoretical characterization of the intuitive idea that if one finds a weight trajectory that transfers from the source to the target task while keeping the generalization gap small at all time instants, then the length of the trajectory is a good indicator of the distance between tasks.

3.4. Experiments

3.4.1. Setup

We use the MNIST, CIFAR-10, CIFAR-100 and Deep Fashion datasets for our experiments. Source and target tasks consist of subsets of these datasets, each task with one or more of the original classes inside it. We show results using an 8-layer convolutional neural network with ReLU nonlinearities, dropout, batch-normalization with a final fully-connected layer along with a larger wide-residual-network WRN-16-4 (Zagoruyko and Komodakis, 2016). Sec. 3.6 gives details about pre-processing, architecture and training.

3.4.2. Baseline methods to estimate task distances

The difficulty of **fine-tuning is the gold standard of distance between tasks**. It is therefore very popular, e.g., Kornblith et al. (2019) use the number of epochs during transfer as the distance. We compute the length of the weight trajectory, i.e., $\int_0^1 |\mathrm{d}w|$ and call this the **fine-tuning distance**. The trajectory is truncated when validation accuracy on the target task is 95% of its final validation accuracy. No transport of the task is performed and the model directly takes SGD updates on the target task after being pre-trained on the source task.

The next baseline is **Task2Vec** (Achille et al., 2019a) which embeds tasks using the diagonal of the FIM of a model trained on them individually. Cosine distance between these vectors is defined as the task distance.

We also compare with the **uncoupled transfer distance** developed in Sec. 3.2.1. This distance computes length of the weight trajectory on the Riemannian distance and also interpolates the data but does not do them synchronously.

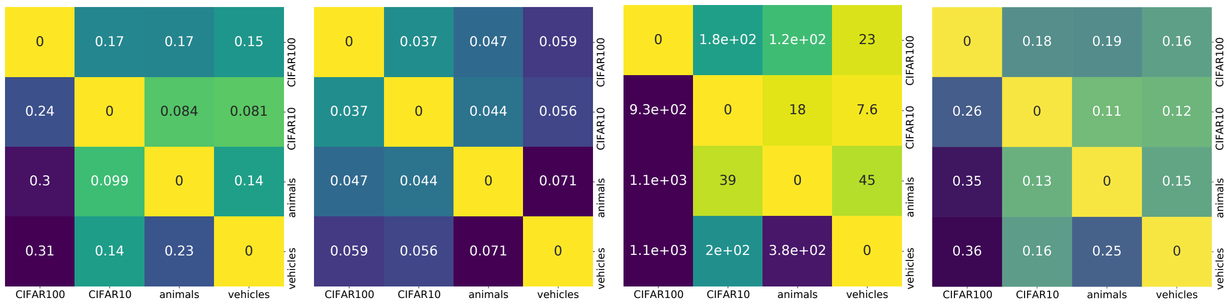
Discrepancy measures on the input space are a popular way to measure task distance. We show task distance computed as the **Wasserstein W_2^2 metric on the the pixel-space**, the **Wasserstein W_2^2 metric on the embedding space** and also method that we devised ourselves where we **transfer a variational autoencoder** (VAE Kingma and Welling (2014)) from the source to the target task and compute the **length of weight trajectory** on the manifold. We transfer the VAE in two ways, (i) by directly fitting the model on the target task, and (ii) by interpolating the task using a mixture distribution as described in Sec. 3.2.1.

3.4.3. Quantitative comparison of distance matrices

Metrics are not unique. We would however still like to compare two task distances across various pairs of tasks. In addition to showing these matrices and drawing qualitative interpretations, we use the Mantel test Mantel (1967) to accept/reject the null hypothesis that variations in two distance matrices are correlated. We will always compute **correlations with the fine-tuning distance matrix** because it is a practically relevant quantity and task distances are often designed to predict this quantity. We report p-values and the normalized test statistic $r = 1/(n^2 - n - 1) \sum_{i,j=1}^{n} (a_{ij} - \bar{a})(b_{ij} - \bar{b})/(\sigma_a \sigma_b)$ where $a, b \in \mathbb{R}^{n \times n}$ are distance matrices for n tasks, $\bar{a}, \sigma_a$ denote mean and standard deviation of entries respectively. Numerical values of r are usually small for all data Ape; Goslee et al. (2007) but the pair (r, p) are a statistically sound way of comparing distance matrices; large r with small p indicates better correlation.

3.4.4. **Transferring between subsets of benchmark datasets**

CIFAR-10 and CIFAR-100 We consider four tasks (i) all vehicles (airplane, automobile, ship, truck) in CIFAR-10, (ii) the remainder, namely six animals in CIFAR-10, (iii) the entire CIFAR-10 dataset and (iv) the entire CIFAR-100 dataset. We show results in Fig. 3.2 using 4×4 distance matrices where numbers in each cell indicate the distance between the source task (row) and the target task (column).



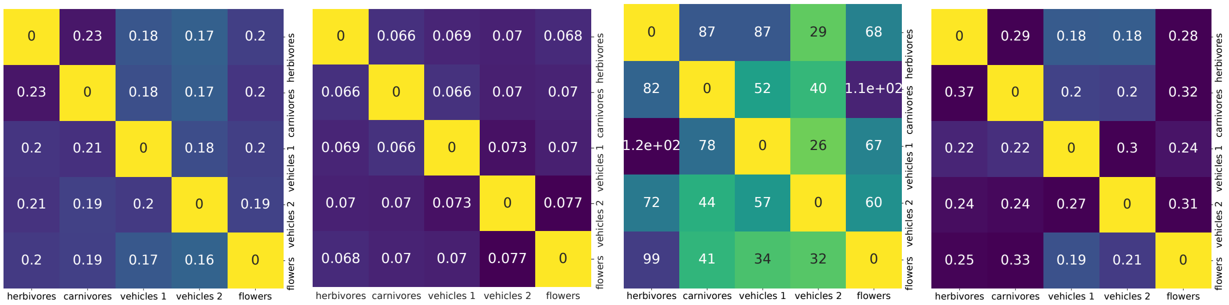
(a) (b) (c) (d)

Figure 3.2: Fig. 3.2a shows coupled transfer distance ($r = 0.428$ $p = 0.13$), Fig. 3.2b shows distances estimated using Task2Vec ($r = 0.03$, $p = 0.98$), Fig. 3.2c shows fine-tuning distance ($r = 0.61$, $p = 0.09$ with itself), and Fig. 3.2d shows uncoupled transfer distance ($r = 0.428$, $p = 0.09$). The numerical values of the distances in this figure are not comparable with each other. Coupled transfer distances satisfy certain sanity checks, e.g., transferring to a subset task is easier than transferring from a subset task (CIFAR-10-vehicles/animals), which Task2Vec does not.

Coupled transfer shows similar trends as fine-tuning, e.g., the tasks animals-CIFAR-10 or vehicles-CIFAR-10 are close to each other while CIFAR-100 is far away from all tasks (it is closer to CIFAR-10 than others). Task distance is asymmetric in Fig. 3.2a, Fig. 3.2c. Distance from CIFAR-10-animals is smaller than animals-CIFAR-10; this is expected because animals is a subset of CIFAR-10. Task2Vec distance estimates in Fig. 3.2b are qualitatively quite different from these two; the distance matrix is symmetric. Also, while fine-tuning from animals-vehicles is relatively easy, Task2Vec estimates the distance between them to be the largest.

This experiment also shows that our approach can scale to medium-scale datasets and can handle situations when the source and target task have different number of classes.

Transferring between subsets of CIFAR-100 We construct five tasks (herbivores, carnivores, vehicles-1, vehicles-2 and flowers) that are subsets of the CIFAR-100 dataset. Each of these tasks consists of 5 sub-classes. The distance matrices for coupled transfer, Task2Vec and fine-tuning are shown in Fig. 3.3a, Fig. 3.3b and Fig. 3.3c respectively. We also show results using uncoupled transfer in Fig. 3.3d.



(a) (b) (c) (d)

Figure 3.3: Fig. 3.3a shows coupled transfer distance ($r = 0.14$, $p = 0.05$), Fig. 3.3b shows Task2Vec distance ($r = 0.07$, $p = 0.17$), Fig. 3.3c shows fine-tuning distance ($r = 0.36$, $p = 0.03$), and Fig. 3.3d shows uncoupled transfer distance ($r = 0.12$, $p = 0.47$). Numerical values in the first and the last sub-plot can be compared directly. Coupled transfer broadly agrees with fine-tuning except for carnivores-flowers and herbivores-vehicles-1. For all tasks, uncoupled transfer overestimates the distances compared to Fig. 3.3a.

Coupled transfer estimates that all these subsets of CIFAR-100 are roughly equally far away from each other with herbivores-carnivores being the farthest apart while vehicles-1-vehicles-2 being closest. This ordering is consistent with the fine-tuning distance although fine-tuning results in an extremely large value for carnivores-flowers and vehicles-1-herbivores. This ordering is mildly inconsistent with the distances reported by Task2Vec in Fig. 3.3b the distance for vehicles-1-vehicles-2 is the highest here. Broadly, Task2Vec also results in a distance matrix that suggests that all tasks are equally far away from each other. As has been reported before (Li et al., 2020), this experiment also demonstrates the fragility of fine-tuning.

Recall that distances for uncoupled transfer in Fig. 3.3d can be compared directly to those in Fig. 3.3a for coupled transfer. Task distances for the former are always larger. Further, distance estimates of uncoupled transfer do not bear much resemblance with those of fine-tuning; see for example the distances for vehicles-2-carnivores, flowers-carnivores, and vehicles-1-vehicles-2. This demonstrates the utility of solving a coupled optimization problem in (3.14) which finds a shorter trajectory on the statistical manifold.

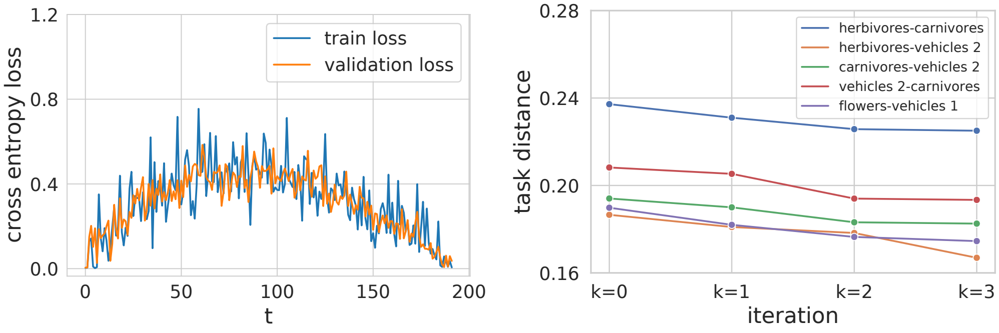
Experiments on **transferring between subsets of Deep Fashion** are given in Sec. 3.6.5. We also computed task distances for tasks with different input domains. For transferring from **MNIST to CIFAR-10**, the coupled transfer distance is 0.18 (0.06 in the other direction), fine-tuning distance is 554.2 (20.6 in the other direction) and Task2Vec distance is 0.149 (same in the other direction). This experiment shows that can robustly handle diverse input domains and yet again, the coupled transfer distance correlates with the fine-tuning distance.

3.4.5. **Further analysis of the coupled transfer distance**

Convergence of coupled transfer Fig. 3.4a shows the evolution of training and test loss as computed on samples of the interpolated distribution after $k = 4$ iterations of (3.14). As predicted by Thm. 20 the generalization gap is small throughout the trajectory. Training loss increases towards the middle; this is expected because the interpolated task is far away from both source and target tasks there. The interpolation (3.17) could also be a cause for this increase.

We typically require 4–5 iterations of (3.14) for the task distance to converge; this is shown in Fig. 3.4b for a few instances. This figure also indicates that computing the transport coupling in (3.2) independently of the weights and using this coupling to modify the weights, as done in say (Cui et al., 2018), results in a larger distance than if one were to optimize the couplings along with the weights. The coupled transfer finds shorter trajectories for weights and will potentially lead to better accuracies on target tasks in studies like (Cui et al., 2018) because it samples more source data.

Models with a larger capacity are easier to transfer We next show that using a model with



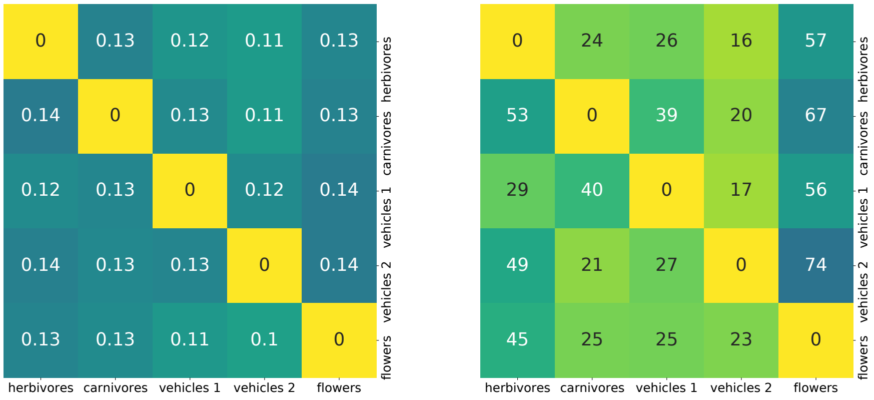
(a) (b)

Figure 3.4: Fig. 3.4a shows the evolution of the training and test cross-entropy loss on the interpolated distribution as a function of the transfer steps in the final iteration of coupled transfer of vehicles-1-vehicles-2. As predicted by Thm. 20, generalization gap along the trajectory is small. Fig. 3.4b shows the convergence of the task distance with the number of iterations k in (3.14); the distance typically converges in 4–5 iterations for these tasks.

higher capacity results in smaller distances between tasks. We consider a wide residual network (WRN-16-4) of Zagoruyko and Komodakis (2016) and compute distances on subsets of CIFAR-100 in Fig. 3.5. First note that task distances for coupled transfer in Fig. 3.5a are consistent with those for fine-tuning in Fig. 3.5b. Coupled transfer distances in Fig. 3.5a are much smaller than those in Fig. 3.3a.

Roughly speaking, a high-capacity model can learn a rich set of features, some discriminative and others redundant not relevant to the source task. These redundant features are useful if target task is dissimilar to the source. This experiment also demonstrates that the information-geometric distance computed by coupled transfer, which is independent of the dimension of the statistical manifold, leads to a constructive strategy for selecting architectures for transfer learning. Most methods to compute task distances instead only inform which source target is best suited to pre-train with for the target task.

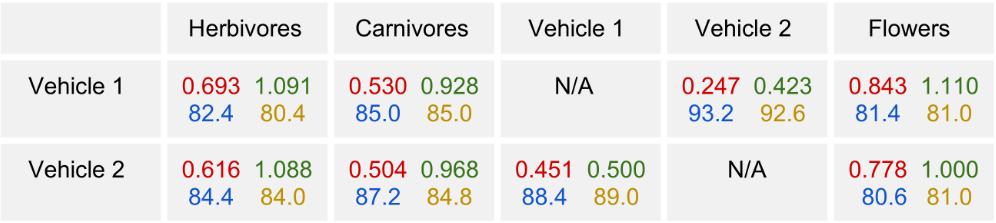
Does coupled transfer lead to better generalization on the target? It is natural to ask whether the generalization performance of the model after coupled transfer is better than the one after standard fine-tuning (which does not transport the task). Fig. 3.6 compares the validation loss and the validation accuracy after coupled transfer and after standard fine-tuning for pairs of



(a) (b)

Figure 3.5: Fig. 3.5a shows coupled transfer distance ($r = 0.15$, $p = 0.01$) and Fig. 3.5b shows fine-tuning distance ($r = 0.39$, $p = 0.01$ with itself and $r = 0.21$, $p = 0.20$ with fine-tuning distance in Fig. 3.3c). Numbers in Fig. 3.5a can be directly compared to those in Fig. 3.3a. WRN-16-4 model has a shorter trajectory for all task pairs compared to the CNN in Fig. 3.3a with fewer parameters.

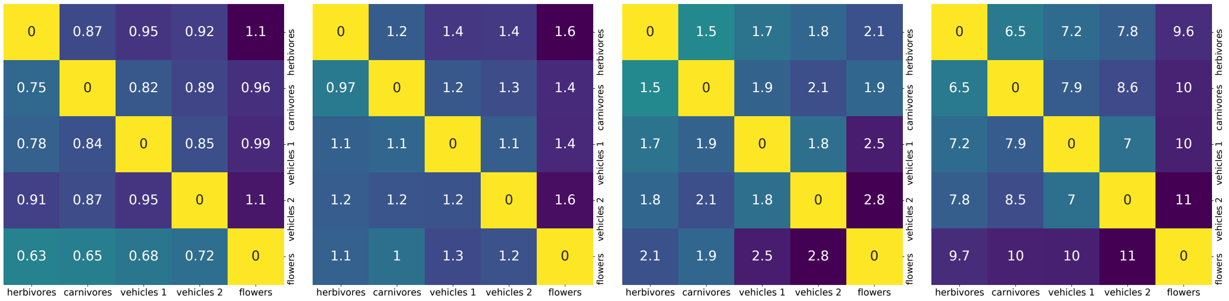
CIFAR-100 tasks. It shows that broadly, the former improves generalization. This is consistent with existing literature Gao and Chaudhari (2020a) which employs task interpolation for better transfer. Let us note that improving fine-tuning is not our goal while developing the task distance. In fact, we want the task distance to correlate with the difficulty of fine-tuning.



	Herbivores	Carnivores	Vehicle 1	Vehicle 2	Flowers
Vehicle 1	0.693 1.091 82.4 80.4	0.530 0.928 85.0 85.0	N/A	0.247 0.423 93.2 92.6	0.843 1.110 81.4 81.0
Vehicle 2	0.616 1.088 84.4 84.0	0.504 0.968 87.2 84.8	0.451 0.500 88.4 89.0	N/A	0.778 1.000 80.6 81.0

Figure 3.6: Comparison of validation loss (red for coupled transfer, green for fine-tuning) and accuracy (%) (blue and yellow respectively) between different subsets of CIFAR-100. Optimal transport for the task distribution results in large improvements in the validation loss in all cases; The validation accuracy also improve by 0.4%–2.5% in all cases except the last two.

Comparison with other task discrepancy measures Fig. 3.7a shows task distances computed using the Riemannian length of the weight trajectory for the VAE (see Sec. 3.4.2) when task is interpolated using a mixture distribution, Fig. 3.7b shows the same quantity when the VAE is directly fitted to the target task after initialization on the source. Fig. 3.7c and Fig. 3.7d show the Wasserstein distance on the pixel-space and feature-space respectively. We find that although the four distance matrices in Fig. 3.7 agree with each other very well ($r \approx 0.15$, $p < 0.08$ for all pairs, except the VAE with uncoupled transfer), they are very different from the fine-tuning distance in Fig. 3.3c. This shows that task distances computed using discrepancy measures on the input space are not reflective of the difficulty of fine-tuning, after all images in these tasks are visually quite similar to each each. Coupled transfer distance explicitly takes the hypothesis space into account and correctly reflects the difficulty of transfer, even if the input spaces are similar.



(a) (b) (c) (d)

Figure 3.7: Fig. 3.7a shows task distance computed using the Riemannian length of the weight trajectory for the VAE using a mixture distribution to interpolate the tasks (see Sec. 3.4.1, $r = 0.1$, $p = 0.76$), Fig. 3.7b shows the same quantity for directly fine-tuning the VAE ($r = 0.09$, $p = 0.88$), Fig. 3.7c shows task distance using the Wasserstein metric on the pixel-space ($r = 0.02$, $p = 0.22$), Fig. 3.7d shows distances using Wasserstein metric on the embedding space ($r = 0.06$, $p = 0.40$). The last three methods agree with each other very well (see the narrative for p-values) but small Mantel test statistic and high p-values as compared to Fig. 3.3c indicates that these distances are not correlated with the difficulty of fine-tuning.

3.5. Related Work and Discussion

Domain-specific methods A rich understanding of task distances has been developed in computer vision, e.g., Zamir et al. (2018) compute pairwise distances when different tasks such as classification, segmentation etc. are performed on the same input data. The goal of this work, and others such as (Cui et al., 2018), is to be able to decide which source data to pre-train to generalize well on a target task. Task distances have also been widely discussed in the multi-task learning (Caruana, 1997) and meta/continual-learning (Liu et al., 2019; Pentina and Lampert, 2014; Hsu et al., 2018). The natural language processing literature also prevents several methods to compute similarity between input data (Mikolov et al., 2013; Pennington et al., 2014).

Most of the above methods are based on evaluating the difficulty of fine-tuning, or computing the similarity in some embedding space. It is difficult to ascertain whether the distances obtained thereby are truly indicative of the difficulty of transfer; fine-tuning hyper-parameters often need to be carefully chosen (Li et al., 2020) and neither is the embedding space unique. For instance, the uncoupled transfer process that modifies the input data distribution will lead to a different estimate of task distance.

Information-theoretic approaches We build upon a line of work that combines generative models and discriminatory classifiers (see (Jaakkola and Haussler, 1999; Perronnin et al., 2010) to name a few) to construct a notion of similarity between input data. Modern variants of this idea include Task2Vec (Achille et al., 2019a) which embeds the task using the diagonal of the FIM and computes distance between tasks using the cosine distance for this embedding. The main hurdle in Task2Vec and similar approaches is to design the architecture for computing FIM: a small model will indicate that tasks are far away. Achille et al. (2019b,c) use the KL divergence between the posterior weight distribution and a prior to quantify the complexity of a task; distance between tasks is defined to be the increase in complexity when the target task is added to the source task. This is an elegant formalism but it is challenging to compute it accurately and it has not yet been demonstrated for a broad range of datasets.

Learning-theoretic approaches Learning theory typically studies out-of-sample performance on a single task using complexity measures such as VC-dimension (Vapnik, 1998). These have been adapted to address the difficulty of domain adaptation (Ben-David et al., 2010; Zhang et al., 2012; Redko et al., 2019) which gives a measure of task distance that incorporates the complexity of the hypothesis space. In particular, Ben-David et al. (2010) train on a fixed mixture of the source and target data to minimize which is similar to our interpolated distribution (3.17). Theoretical results here corroborate (actually motivate) our experimental result that transferring between the same tasks with a higher-capacity model is easer. A key gap in this literature is that this theory does not consider *how* the model is adapted to target task. For complex models such as deep networks, hyper-parameters during fine-tuning play a crucial role (Li et al., 2020). Our work fundamentally exploits the idea that the task need not be fixed during transfer, it can also be adapted. Further, our coupled transfer distance is invariant to the particular parametrization of the deep network, which is difficult to achieve using classical learning theory techniques.

Coupled transfer of data and the model Transporting the task using optimal transport is fundamental to how our coupled transfer distance is defined. This is motivated from two recent studies. Gao and Chaudhari (2020a) develop an algorithm that keeps the classification loss unchanged across transfer. Their method interpolates between the source and target data using the mixture distribution from Sec. 3.2.1. We take this idea further and employ optimal transport Cui et al. (2018) to modulate the interpolation of the task using the Fisher-Rao distance. Coupled transport problems on the input data are also solved for unsupervised translation (Alvarez-Melis and Jaakkola, 2018). The idea of modifying the task during transfer using optimal transport is also exploited by Alvarez-Melis and Fusi (2020a) to prescribe task distances and for data augmentation/interpolation and transfer (Alvarez-Melis and Fusi, 2020b).

Our work is an attempt to theoretically understand when transfer is easy and when it is not. An often over-looked idea in large-scale transfer learning is that the task need not remain fixed to the target task during transfer. We heavily exploit this idea in the present paper. We develop a "coupled transfer distance" between tasks that computes the shortest weight trajectory in information space, i.e., on the statistical manifold, while the task is optimally transported from the source to the target. The most important aspect of our work is that both task and weights are modified synchronously. It is remarkable that this coupled transfer distance is not just strongly correlated with the difficulty of fine-tuning but also theoretically captures the intuitive idea that a good transfer algorithm is the one that keeps generalization gap small during transfer, in particular at the end on the target task.

3.6. Appendix

3.6.1. Architecture and training

We show results using an 8-layer convolutional neural network with ReLU nonlinearities, dropout, batch-normalization with a final fully-connected layer. The larger model used for experiments in Fig. 3.5 is a wide-residual-network (WRN-16-4 architecture of (Zagoruyko and Komodakis, 2016)).

3.6.2. Transferring between CIFAR-10 and CIFAR-100

We consider four tasks: (i) all vehicles (airplane, automobile, ship, truck) in CIFAR-10, consisting of 20,000 32×32-sized RGB images; (ii) the remainder, namely six animals in CIFAR-10, consisting of 30,000 32×32-sized RGB images; (iii) the entire CIFAR-10 dataset and (iv) the entire CIFAR-100 dataset, consisting of 50,000 images and spread across 100 classes.

We pre-train model on source tasks using stochastic gradient descent (SGD) for 60 epochs, with mini-batch size of 20, learning rate schedule is set to 10^{-3} for epochs $0-40$ and 8×10^{-4} for epochs $40-60$. When CIFAR-100 is the source dataset, we train for 180 epochs with the learning rate set to 10^{-3} for epochs $0-120$, and 8×10^{-4} for epochs $120-180$.

We chose a slightly smaller version of the source and target datasets to compute the distance, each of them have 19,200 images. The class distribution on all source and target classes is balanced. We did this to reduce the size of the coupling matrix Γ in (3.14). The coupling matrix connecting inputs in the source and target datasets is $\Gamma \in \mathbb{R}^{19200\times 19200}$ which is still quite large to be tractable during optimization. We therefore use a block diagonal approximation of the coupling matrix; 640 blocks are constructed each of size 30×30 and all other entries in the coupling matrix are set to zero at the beginning of each iteration in (3.14) after computing the dense coupling matrix using the linear program. This effectively entails that the set of couplings over which we compute the transport is not the full convex polytope in Sec. 3.1.1 but rather a subset of it. We sample a mini-batch of 20 images from the interpolated distribution corresponding to this block-diagonal coupling matrix for each weight update of (3.16). We run 40 epochs, i.e., with $19200/20 = 960$ weight updates per epoch for computing the weight trajectory at *each iteration* k in (3.14). The learning rate is fixed to 8×10^{-4} in the transfer learning phase.

3.6.3. Transferring among subsets of CIFAR-100

The same 8-layer convolutional network is used to show results for transfer between subsets of CIFAR-10 and CIFAR-100. CIFAR-10 is split into the two tasks animals and vehicle again. We construct five tasks (herbivores, carnivores, vehicles-1,vehicles-2 and flowers) that are subsets of the CIFAR-100 dataset. Each of these tasks consists of 5 sub-classes.

We train the model on the source task using SGD for 400 epochs with a mini-batch size of 20. Learning rate is set to 10^{-3} for epochs $0 - 240$, and to 8×10^{-4} for epochs $240 - 400$.

Tasks that are subsets of CIFAR-100 in the experiments in this section have few samples (2500 each) so we select 2400 images from source and target datasets respectively; we could have chosen a larger source dataset when transferring from CIFAR-10 animals or vehicles but we did not so for sake of simplicity. The number 2400 was chosen to make the block diagonal approximation of the coupling matrix have 120×120 entries in each block; this was constrained by the GPU memory. The coupling matrix Γ therefore has 2400×2400 entries with 20 blocks on the diagonal.

Again, we use a mini-batch size of 20 for 240 epochs ($2400/20 = 120$ weight updates per epoch) during the transfer from the source dataset to the target dataset. The learning rate is fixed to 8×10^{-4} in the transfer learning phase.

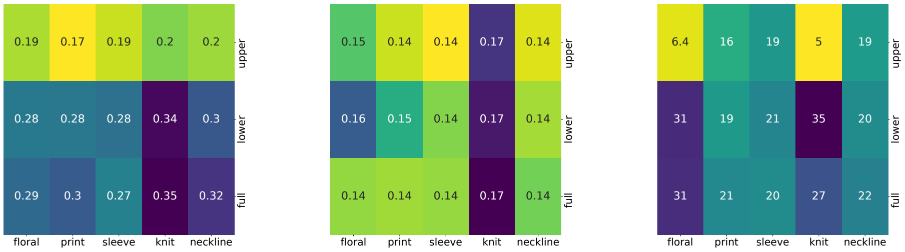
3.6.4. Training setup for wide residual network

We pre-train WRN-16-4 on source tasks using SGD for 400 epochs with a mini-batch size of 20. Learning rate is 10^{-1} for epochs $0 - 120$, 2×10^{-2} for epochs $120 - 240$, 4×10^{-3} for epochs 240–320, and 8×10^{-4} for epochs $320 - 400$. Other experimental details are the same as those in Sec. 3.6.3.

3.6.5. Experiments on the Deep Fashion dataset

For the Deep Fashion dataset (Liu et al., 2016), we consider three binary category classification tasks (upper clothes, lower clothes, and full clothes) and five binary attribute classification tasks (floral, print, sleeve, knit, and neckline). We show results in Fig. 3.8 using 3× 5 distance matrices where numbers in each cell indicate the distance between the source task (row) and the target task (column). We show results using a wide-residual-network (WRN-16-4, (Zagoruyko and Komodakis, 2016)).

The model is trained using SGD for 400 epochs with a mini-batch size 20. Learning rate is 10^{-1} for



(a) (b) (c)

Figure 3.8: Fig. 3.8a shows distances (numbers in the cell) among sub-tasks in DeepFashion computed using our coupled transfer process (r = 0.37, p = 0.33), Fig. 3.8c shows distances estimated using Task2Vec (r = 0.04, p = 0.75) while Fig. 3.8c shows distances estimated using fine-tuning (r = 0.54, p = 0.36 with itself). Numerical values of the distances in this figure are not comparable with each other. Coupled transfer, Task2Vec and fine-tuning all agree with that transferring to knit is relatively hard. Transferring from upper-cloth to knit is easy via fine-tuning and coupled transfer correctly estimates this distance to be small; the distance estimated by Task2Vec is much larger in comparison. Since these matrices are non-square, we ran the Mantel test for three 3×3 submatrices (sweep across columns) of these 3×5 matrices and report the mean test statistic and the average p-value across these tests above.

epochs 0 – 120, 2×10^{-2} for epochs 120 – 240, 4×10^{-3} for epochs 240–320, and 8×10^{-4} for epochs 320 – 400. We sample 14,000 images from the source and target datasets to compute distances. A mini-batch size of 20 is used during transfer and we run (3.16) for 60 epochs (14000/20 = 700 weight updates per epoch).

3.6.6. Proof of Thm. 20

We first prove a simpler theorem.

Theorem 21. Given a trajectory of the weights $\{w(\tau)\}_{\tau \in [0,1]}$ and a sequence $0 \leq \tau_1 < \tau_2 < ... < \tau_K \leq 1$, then for all $\epsilon > \frac{2}{K}\sum_{k=1}^{K} \mathcal{R}_N(\|w(\tau_k)\|_{\text{FR}})$, the probability that

$$\frac{1}{K}\sum_{k=1}^{K}\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}}\left[\ell(\omega(\tau_k), x, y)\right] - \frac{1}{N}\sum_{(x,y)\sim \hat{p}_{\tau_k}} \ell(\omega(\tau_k), x, y)\right)$$

is greater than ϵ is upper bounded by

$$\exp\left\{-\frac{2K}{M^2}\left(\epsilon - \frac{2}{K}\sum_{k=1}^{K}\mathcal{R}_N(\|w(\tau_k)\|_{\text{FR}})\right)^2\right\}. \tag{3.20}$$

Proof. For each moment τ_k, by taking supremum

$$\begin{aligned}
&\mathbb{E}_{(x,y)\sim p_{\tau_k}}\ell(w(\tau_k),x,y)-\frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}}\ell(w(\tau_k),x,y)\\
\leq&\sup_{\|w\|_{\mathrm{FR}}\leq\|w(\tau_k)\|_{\mathrm{FR}}}\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}}\ell(w,x,y)-\frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}}\ell(w,x,y)\right),
\end{aligned}\tag{3.21}$$

where $\|\cdot\|_{\mathrm{FR}}$ denotes Fisher-Rao norm (Liang et al., 2019). The right hand side of inequality(3.21) is a random variable that depends on the drawn sampling set $\hat{p}_{\tau_k}$ with size N. Denoting

$$\varphi(\hat{p}_{\tau_k}):=\sup_{\|w\|_{\mathrm{FR}}\leq\|w(\tau_k)\|_{\mathrm{FR}}}\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}}\ell(w,x,y)-\frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}}\ell(w,x,y)\right),\tag{3.22}$$

We would like to bound the expectation of $\phi(\hat{p}_{\tau_k})$ in terms of the Rademacher complexity. In order to do this, we introduce a "ghost sample" with size N, $\hat{p}'_{\tau_k}$, independently drawn identically from $p_{\tau_k}(x,y)$, we rewrite the expectations

$$\begin{aligned}
\mathbb{E}_{\hat{p}_{\tau_k}}\phi(\hat{p}_{\tau_k})&=\mathbb{E}_{\hat{p}_{\tau_k}}\left[\sup_{\|w\|_{\mathrm{FR}}\leq\|w(\tau_k)\|_{\mathrm{FR}}}\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}}\ell(w,x,y)-\frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}}\ell(w,x,y)\right)\right]\\
&=\mathbb{E}_{\hat{p}_{\tau_k}}\left[\sup_{\|w\|_{\mathrm{FR}}\leq\|w(\tau_k)\|_{\mathrm{FR}}}\mathbb{E}_{\hat{p}'_{\tau_k}}\left(\frac{1}{N}\sum_{(x,y)\sim\hat{p}'_{\tau_k}}\ell(w,x,y)-\frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}}\ell(w,x,y)\right)\right]\\
&\leq\mathbb{E}_{\hat{p}_{\tau_k},\hat{p}'_{\tau_k},\sigma}\left[\sup_{\|w\|_{\mathrm{FR}}\leq\|w(\tau_k)\|_{\mathrm{FR}}}\frac{1}{N}\left(\sum_{(x,y)\sim\hat{p}_{\tau_k}}\sigma^i(\ell(w,x,y)-\ell(w,x,y))\right)\right]\\
&\leq\mathbb{E}_{\hat{p}_{\tau_k},\sigma}\left[\sup_{\|w\|_{\mathrm{FR}}\leq\|w(\tau_k)\|_{\mathrm{FR}}}\frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}}\sigma^i\ell(w,x,y)\right]\\
&+\mathbb{E}_{\hat{p}_{\tau_k},\sigma}\left[\sup_{\|w\|_{\mathrm{FR}}\leq\|w(\tau_k)\|_{\mathrm{FR}}}\frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}}\sigma^i\ell(w,x,y)\right]\\
&=2\mathcal{R}_N(\|w(\tau_k)\|_{\mathrm{FR}}),
\end{aligned}$$

where $\sigma = (\sigma^1, \sigma^2, \dots, \sigma^N)$ are independent random variables drawn from the Rademacher distribution, the last equality is followed by the definition of Rademacher Complexity within $\|w(\tau_k)\|_{\text{FR}}$-ball in the Fisher-Rao norm. By Hoeffding's lemma, for $\lambda > 0$

$$\begin{aligned}\mathbb{E}_{\hat{p}_{\tau_k}} \exp\left\{\lambda\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}} \ell(w(\tau_k), x, y) - \frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}} \ell(w(\tau_k), x, y)\right)\right\} &= \mathbb{E}_{\hat{p}_{\tau_k}} e^{\lambda\phi(\hat{p}_{\tau_k})} \\ &\le e^{\lambda\mathbb{E}_{\hat{p}_{\tau_k}}\phi(\hat{p}_{\tau_k}) + \frac{\lambda^2 M^2}{8}} \\ &\le e^{2\lambda\mathcal{R}_N(\|w(\tau_k)\|_{\text{FR}}) + \frac{\lambda^2 M^2}{8}}.\end{aligned} \tag{3.23}$$

For each moment τ_k, we have inequality(3.23), which implies

$$\begin{aligned}&\mathbb{E}_{\hat{p}_{\tau_k}:\ 1\le k\le K} \exp\left\{\lambda\sum_{k=1}^{K}\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}} \ell(w(\tau_k), x, y) - \frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}} \ell(w(\tau_k), x, y)\right)\right\} \\ &= \prod_{k=1}^{K}\mathbb{E}_{\hat{p}_{\tau_k}} \exp\left\{\lambda\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}} \ell(w(\tau_k), x, y) - \frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}} \ell(w(\tau_k), x, y)\right)\right\} \\ &\le \exp\left\{\sum_{k=1}^{K}\left[2\lambda\mathcal{R}_N(\|w(\tau_k)\|_{\text{FR}}) + \frac{\lambda^2 M^2}{8}\right]\right\}.\end{aligned}$$

Finally for all $K\epsilon > 2\sum_{k=1}^{K}\mathcal{R}_N(\|w(\tau_k)\|_{\text{FR}})$, by Markov's inequality

$$\begin{aligned}&Pr\left\{\sum_{k=1}^{K}\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}} \ell(w(\tau_k), x, y) - \frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}} \ell(w(\tau_k), x, y)\right) > K\epsilon\right\} \\ &\le \exp\left\{-\lambda K\epsilon + \sum_{k=1}^{K}\left[2\lambda\mathcal{R}_N(\|w(\tau_k)\|_{\text{FR}}) + \frac{\lambda^2 M^2}{8}\right]\right\}\end{aligned} \tag{3.24}$$

Put $\lambda = \frac{4K\left(\epsilon - \frac{2}{K}\sum_{k=1}^{K}\mathcal{R}_N(\|w(\tau_k)\|_{\text{FR}})\right)}{M^2}$ in right hand side of inequality(3.24), then we finish the proof.

□

3.6.7. **Proof of Thm. 20**

The upper bound in (3.24) above states that we should minimize the Rademacher complexity of the hypothesis space in order to ensure that the weight trajectory has a small generalization gap at all time instants. For linear models, as discussed in the main paper (Liang et al., 2019), the Rademacher complexity can be related to the Fisher-Rao norm $\langle w, gw \rangle$. The Fisher-Rao distance on the manifold, namely

$$\int_0^1 \mathbb{E}_{x \sim p_\tau(x)} \left[\sqrt{2\text{KL}\left(p_{w(\tau)}(\cdot|x),\ p_{w(\tau+\mathrm{d}\tau)}(\cdot|x)\right)} \right] \mathrm{d}\tau = \int_0^1 \mathbb{E}_{x \sim p_\tau(x)} \sqrt{\left\langle \dot{w(\tau)}, g(w(\tau)) \dot{w(\tau)} \right\rangle}\ \mathrm{d}\tau \tag{3.25}$$

is only a lower bound on the integral of the Fisher-Rao norm along the weight trajectory. We therefore make some additional assumptions in this section to draw out a crisp link between the Fisher-Rao *distance* and generalization gap along the trajectory.

Let $\ell(w; x, y) = -\log p_w(y \,|\, x)$ be the cross-entropy loss on sample (x, y). We assume that at each moment $\tau \in [0, 1]$, our model $p_{w(\tau)}(y \,|\, x)$ predicts on the interpolating distribution $p_\tau(y \,|\, x)$ well, that is

$$p_{w(\tau)}(y \,|\, x) \approx p_\tau(y \,|\, x)$$

for all input x; this is a reasonable assumption and corresponds to taking a large number of mini-batch updates in (3.16). We approximate the FIM using the empirical FIM, i.e., we approximate the distribution $p_\tau(y|x)$ as a Dirac-delta distribution on the interpolated labels $y_\tau(x)$. Observe that

$$\begin{aligned} \left\langle \dot{w(\tau)}, g(w(\tau)) \dot{w(\tau)} \right\rangle &= \left\langle \dot{w(\tau)}, \mathbb{E}_{y \,|\, x \sim p_\tau} \partial_w \ell_{w(\tau)}(y \,|\, x) \partial_w \ell_{w(\tau)}(y \,|\, x)^\top \dot{w(\tau)} \right\rangle \\ &\approx \left\langle \dot{w(\tau)}, \partial_w \ell(w(\tau); x, y_\tau(x))\ \partial_w \ell(w(\tau); x, y_\tau(x))^\top \dot{w(\tau)} \right\rangle \\ &= \left| \frac{\ell(w(\tau + \mathrm{d}\tau); x, y_\tau(x)) - \ell(w(\tau); x, y_\tau(x))}{\mathrm{d}\tau} \right|^2 \\ &= \left| \frac{\Delta \ell(w(\tau))}{\mathrm{d}\tau} \right|^2, \end{aligned} \tag{3.26}$$

where we use the shorthand

$$\Delta \ell(w(\tau)) := \ell(w(\tau + \mathrm{d}\tau); x, y_\tau(x)) - \ell(w(\tau); x, y_\tau(x)),$$

and plug (3.26) in the integration in (3.25)

$$\begin{aligned}\int_0^1 \mathbb{E}_{x\sim p_\tau(x)}\left[\sqrt{2\mathrm{KL}\left(p_{w(\tau)}(\cdot|x),\ p_{w(\tau+\mathrm{d}\tau)}(\cdot|x)\right)}\right]\mathrm{d}\tau &= \int_0^1 \mathbb{E}_{x\sim p_\tau(x)}\sqrt{\left\langle \dot{w(\tau)}, g(w(\tau))\dot{w(\tau)}\right\rangle}\,\mathrm{d}\tau \\ &\approx \int_0^1 \mathbb{E}_{x\sim p_\tau(x)}\left[|\Delta\ell(w(\tau)|\right].\end{aligned} \tag{3.27}$$

On the other hand, for moment τ let $\Omega_\tau \ni w(\tau)$ be a compact neighborhood of $w(\tau)$ in weights space, Rademacher complexity of the class of loss function is upper bounded as following

$$\begin{aligned}\mathcal{R}_N(\Omega_\tau) &= \mathbb{E}_{\hat{p}\sim p_\tau^N}\mathbb{E}_\sigma\left[\sup_{w\in\Omega_\tau}\frac{1}{N}\sum_{i=1}^N \sigma^i\ell(w;x^i,y^i)\right] \\ &= \mathbb{E}_{\hat{p}\sim p_\tau^N}\mathbb{E}_\sigma\left[\sup_{w\in\Omega_\tau}\frac{1}{N}\sum_{i=1}^N \sigma^i\ell(w(\tau);x^i,y^i)+\sigma^i\left(\ell(w;x^i,y^i)-\ell(w(\tau);x^i,y^i)\right)\right] \\ &\leq \mathbb{E}_{\hat{p}\sim p_\tau^N}\mathbb{E}_\sigma\left[\frac{1}{N}\sum_{i=1}^N \sigma^i\ell(w(\tau);x^i,y^i)+\sup_{w\in\Omega_\tau}\frac{1}{N}\sum_{i=1}^N |\ell(w;x^i,y^i)-\ell(w(\tau);x^i,y^i)|\right], \\ &= 0+\mathbb{E}_{\hat{p}\sim p_\tau^N}\left[\sup_{w\in\Omega_\tau}\frac{1}{N}\sum_{i=1}^N |\ell(w;x^i,y^i)-\ell(w(\tau);x^i,y^i)|\right] \\ &\longrightarrow \sup_{w\in\Omega_\tau}\mathbb{E}_{x\sim p_\tau}|\ell(w;x,y_\tau(x))-\ell(w(\tau);x,y_\tau(x))|\end{aligned} \tag{3.28}$$

as N goes to infinity. The last step in (3.28) is followed by the compactness of Ω_τ and the Lipschitz continuity of the loss function. Let

$$\Omega_\tau := \{w|\mathbb{E}_{x\sim p_\tau}|\ell(w;x,y_\tau(x))-\ell(w(\tau);x,y_\tau(x))| \leq \mathbb{E}_{x\sim p_\tau}|\ell(w(\tau+\mathrm{d}\tau);x,y_\tau(x))-\ell(w(\tau);x,y_\tau(x))|\}, \tag{3.29}$$

be the neighborhood of $w(\tau)$ within which the loss function changes less than $|\Delta\ell(w(\tau))|$. Compare this with (3.27), the Rademacher complexity of Ω_τ is exactly upper bounded by integration increments appearing in the expression for the Fisher-Rao distance. If we substitute $\|w(\tau)\|_{\mathrm{FR}}$-ball in (3.20) with this modified Ω_τ, we have the following theorem.

Theorem 22. Given a trajectory of the weights $\{w(\tau)\}_{\tau\in[0,1]}$ and a sequence $0=\tau_0\leq\tau_1<\tau_2<$

... $< \tau_K \leq 1$, for all $\epsilon > 2\sum_{k=1}^{K}(\tau_k - \tau_{k-1})\mathbb{E}_{x\sim p_\tau}|\Delta\ell(w(\tau_{k-1}))|$, the probability that

$$\frac{1}{K}\sum_{k=1}^{K}\left(\mathbb{E}_{(x,y)\sim p_{\tau_k}}\left[\ell(\omega(\tau_k),x,y)\right] - \frac{1}{N}\sum_{(x,y)\sim\hat{p}_{\tau_k}}\ell(\omega(\tau_k),x,y)\right)$$

is greater than ϵ is upper bounded by

$$\exp\left\{-\frac{2K}{M^2}\left(\epsilon - 2\sum_{k=1}^{K}(\tau_k - \tau_{k-1})\mathbb{E}_{x\sim p_{\tau_k}}\left[|\Delta\ell(w(\tau_{k-1}))|\right]\right)\right\}. \tag{3.30}$$

Proof. The proof is same as in (3.20) except for substituting $\mathcal{R}_N(\|w(\tau_k)\|_{\mathrm{FR}})$ with $\mathcal{R}_N(\Omega_{\tau_k})$ and using upper bounds (3.28), and

$$\begin{aligned}\Omega_{\tau_k} &= \{w|\mathbb{E}_{x\sim p_{\tau_k}}|\ell(w;x,y_{\tau_k}(x)) - \ell(w(\tau_k);x,y_{\tau_k}(x))| \\ &\leq K(\tau_k - \tau_{k-1})\mathbb{E}_{x\sim p_{\tau_k}}|\ell(w(\tau_k);x,y_{\tau_k}(x)) - \ell(w(\tau_{k-1});x,y_{\tau_k}(x))|\}.\end{aligned} \tag{3.31}$$

□

We can now relate the Fisher-Rao distance (3.25) and the generalization bound in Thm. 22. For instance, if $\left|\frac{\mathrm{d}}{\mathrm{d}\tau}\ell(w(\tau);x,y_\tau(x))\right|$ is Riemann integrable over τ, then as K goes to infinity, there exists a sequence $0 = \tau_0 \leq \tau_1 < \tau_2 < ... < \tau_K \leq 1$ such that

$$\begin{aligned}&\sum_{k=1}^{K}(\tau_k - \tau_{k-1})\mathbb{E}_{x\sim p_{\tau_k}}\Big|\ell(w(\tau_k);x,y_{\tau_k}(x)) - \ell(w(\tau_{k-1});x,y_{\tau_k}(x))\Big| \\ &\longrightarrow \int_0^1 \mathbb{E}_{x\sim p_\tau(x)}\left|\ell(w(\tau+\mathrm{d}\tau);x,y_\tau(x)) - \ell(w(\tau);x,y_\tau(x))\right| \\ &\approx \int_0^1 \mathbb{E}_{x\sim p_\tau(x)}\left[\sqrt{2\mathrm{KL}\left(p_{w(\tau)}(\cdot|x),\ p_{w(\tau+\mathrm{d}\tau)}(\cdot|x)\right)}\right]\mathrm{d}\tau.\end{aligned} \tag{3.32}$$

This shows that computing the Fisher-Rao distance between two points on the statistical manifold results in a weight trajectory that minimizes the the generalization gap of weights trained on the interpolated distribution along the trajectory. In other words, one may either think of our coupled transfer process as computing the Fisher-Rao distance or as finding a weight trajectory that connects weights with a small generalization gap.

CHAPTER 4

A CANONICAL APPROACH FOR PRE-TRAINING WITH UNLABELED DATA

The free energy principle highlights the effectiveness of reconstruction (2.5) in pre-training, whereas an auto-encoder is used to reconstruct the original input data x using the pre-trained representation z. In the context of pre-training with unlabeled data $p(x)$, we can also harness the power of reconstruction. Successful algorithms in self-supervised learning (e.g., SimCLR) intuitively design the tasks to pre-train the models using the unlabelled data (e.g., representations invariant to the data augmentations) prior to knowing the actual downstream task $p(x, y)$.

Instead of artificially designing tasks, a canonical approach to explore unlabeled data is to directly reconstruct the potential downstream tasks $p(x, y)$ based on inputs x. Given the input data distribution $p(x)$, the potential tasks that correlate with it are not unique. For instance, an input image can be labeled based on various factors such as background colors, textures, and objects present within it. By leveraging the power of a *reference prior*, we reconstruct a pool of diverse tasks that encompass the typical downstream tasks without knowing the actual labels. These techniques provide us with deeper insights into the space of tasks, as we will discuss in detail later on Sec. 4.2.3.

4.1. Backgrounds

Consider a labelled dataset $\{(x_i, \hat{y}_i)\}_{i=1}^{K}$ with K samples that consists of inputs $x_i \in \mathcal{X}$ and ground-truth labels $\hat{y}_i \in \{1, \ldots, C\}$. Each sample of this dataset is drawn from the given task. We will use the shorthand $X^K = \{x_1, \ldots, x_K\}$ and $\hat{Y}^K = \{\hat{y}_1, \ldots, \hat{y}_K\}$ to denote all inputs and ground truth labels. Let $w \in \mathbb{R}^p$ be the weights of a network that evaluates the conditional probability $p(y \mid x, w)$; here, y is not necessarily a ground truth label. We will use random variables $z = (x, y)$ and $Z^K = (X^K, Y^K)$, with a probability distribution $p(z \mid w)$, when we do not wish to distinguish between inputs and labels.

Given a prior on weights $\pi(w)$, Bayes law gives the posterior

$$p(w \mid X^K, \hat{Y}^K) \propto p(\hat{Y}^K \mid X^K, w)\pi(w).$$

Where $p(\hat{Y}^K \mid X^K, w) = \prod_{i=1}^K p(y_i \mid x_i, w)$, since we assume data are independently and identically sampled from the task domain. The Fisher Information Matrix (FIM) $g \in \mathbb{R}^{p \times p}$ has entries $g(w)_{kl} =$

$$\frac{1}{K} \sum_{i=1}^{K} \sum_{y=1}^{C} p(y \mid x_i, w) \partial_{w_k} \log p(y \mid x_i, w) \partial_{w_l} \log p(y \mid x_i, w),$$

where $1 \leq k, l \leq p$. It can be used to define the *Jeffrey's prior*

$$\pi_J(w) \propto \sqrt{\det g(w)}. \tag{4.1}$$

Jeffrey's prior is reparameterization invariant, i.e., it assigns the same probability to a set of models irrespective of our choice of parameterization of those models. It is an uninformative prior, e.g., it imposes some generic structure on the problem (reparameterization invariance).

4.1.1. Reference priors

To make the choice of a prior more objective, Bernardo (1979) suggested that uninformative priors should maximize some divergence, say the Kullback-Leibler (KL) divergence

$$\mathrm{KL}(p(w \mid Z), \pi(w)) = \int d(x|z) w p(w \mid Z) \log \frac{p(w \mid Z)}{\pi(w)},$$

between the prior $\pi(w)$ and the posterior after seeing infinite observations $Z = \{z_1,\ z_2, ...\}$ ($p(w \mid Z)$). The rationale for doing so is to allow the data to dominate the posterior rather than our choice of the prior. Since we do not know the data *a priori* while picking the prior, we should maximize the *average* KL divergence over the data distribution. This amounts to maximizing the mutual information $I_\pi(w; Z)$

$$\begin{aligned}\pi^* :&= \underset{\pi}{\operatorname{argmax}} \int d(x|z)Zp_\pi(Z) \int d(x|z)wp_\pi(w \,|\, Z) \log \frac{p_\pi(w \,|\, Z)}{\pi(w)} \\ &= \underset{\pi}{\operatorname{argmax}}\, I_\pi(w; Z) \\ &= \underset{\pi}{\operatorname{argmax}}\, H_\pi(w) - H_\pi(w \,|\, Z)\end{aligned} \tag{4.2}$$

where $p_\pi(Z) = \int d(x|z)w\pi(w)p(Z \,|\, w)$ and $\pi(w)p(w \,|\, Z) = p_\pi(Z)p_\pi(w \,|\, Z)$.

$$H_\pi(w) = -\int d(x|z)w\pi(w) \log \pi(w)$$

is the Shannon entropy; the conditional entropy $H_\pi(w \,|\, Z)$ is defined analogously. Mutual information is a natural quantity for measuring the amount of missing information about w provided by infinite observations Z if the initial belief was π. The prior $\pi^*(w)$ is known as a reference prior. It is invariant to a reparameterization of the weight space because mutual information is invariant to reparameterization. The reference prior does not depend upon the samples but only depends on their distribution.

The objective to calculate reference prior π^* above may not be analytically tractable, and therefore Bernardo also suggested computing n-reference priors. We call n the "order" and deliberately use a different notation with the number of samples; we would like to emphasize that those are different concepts, and the reason will be clear soon. $n-$reference prior π_n^* maximize the mutual information between the $n-$replica observations Z^n and the model parameterization w,

$$\begin{aligned}\pi_n^* &= \underset{\pi}{\operatorname{argmax}}\, I_\pi(w; Z^n) \\ &= \underset{\pi}{\operatorname{argmax}} \int d(x|z)wd(x|z)Z^n\pi(w)p(Z^n \,|\, w) \log \frac{p(Z^n \,|\, w)}{p_\pi(Z^n)},\end{aligned} \tag{4.3}$$

where $p_\pi(Z^n) = \int d(x|z)w\pi(w)p(Z^n \,|\, w)$. One may set $\pi^* := \lim_{n\to\infty} \pi_n^*$ under appropriate technical conditions (Berger et al., 1988). Reference priors are asymptotically equivalent to Jeffrey's prior for one-dimensional problems. In general, they differ for multi-dimensional problems, but it can be shown that Jeffrey's prior is the continuous prior that maximizes the mutual information (Clarke and Barron, 1994).

4.2. Reference Priors Reconstruct Tasks at the Boundaries of the Hypothesis Class

Given a $C-$ way classification task with input domain ($x \in \mathcal{X}$) and a family of neural networks parameterized by $w \in \mathbb{R}^p$, the *hypothesis class* $\mathcal{H}$ is a collection of candidate solutions that can be used to predict the labels. Each candidate solution in this hypothesis class is a probability distribution $p_w(y \,|\, x)$ parameterized by w. The hypothesis class in this chapter consists of the probability distributions p_w with the same network architecture but different combinations of parameters $w \in \mathbb{R}^p$.

In this section, we first use a few examples (Sec. 4.2.1 and Sec. 4.2.2) to illustrate how an idea from Bayesian statistics, the n-reference prior (4.3), reconstructs the tasks at the boundaries of the hypothesis class.

4.2.1. First example: estimating the bias of a coin

Blahut-Arimoto algorithm The Blahut-Arimoto algorithm (Arimoto, 1972; Blahut, 1972) is a method for maximizing functionals like (4.3) and leads to iterations t of the form

$$\pi^{t+1}(w) \propto \exp\left(\mathrm{KL}(p(Z^n \,|\, w), p_\pi(Z^n))\right) \pi^t(w).$$

It is typically implemented for discrete variables, e.g., in the Information Bottleneck (Tishby et al., 1999). In this case, maximizing mutual information is a convex problem; therefore, the BA algorithm is guaranteed to converge. Such discretization is difficult for high-dimensional deep networks. We, therefore, implement the BA algorithm using particles; see Rem. 23.

To ground intuition, consider the estimation of the bias of a coin $w \in [0, 1]$ using n trials. Let Z^n denote the sequence of heads or tails we observe(which is a sufficient statistic). For $n = 1$, since we know that $I(w; z^1) \leq \log 2$ with this one bit of information, we can see that $\pi_1^*(z) = (\delta(w) + \delta(1-w))/2$ is the reference prior that achieves this upper bound. This result is intuitive:

if we *know* that we have only one observation, then the optimal uninformative prior should put equal probability mass on the two exhaustive outcomes $w = 0$ (heads) and $w = 1$ (tails). We can numerically calculate π_n^* for different values of n using the BA algorithm (Fig. 4.1).

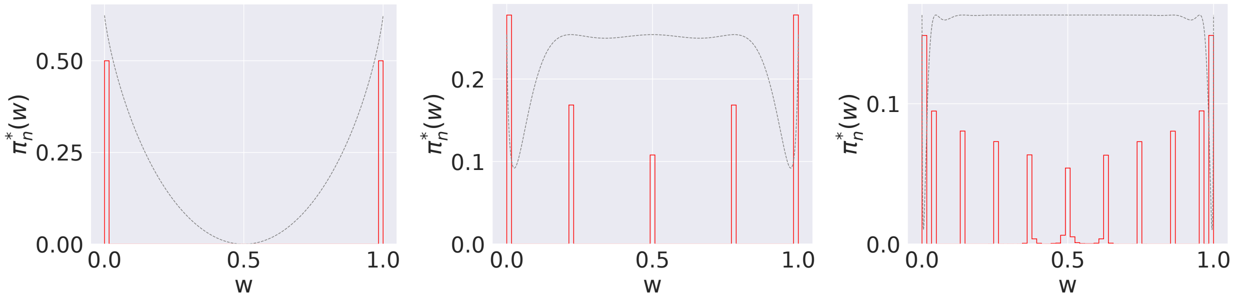


Figure 4.1: We calculated the probability density function of $n-$**reference priors for the coin-tossing model** for $n = 1, 10, 50$ (from left to right) using the Blahut-Arimoto algorithm. Atoms are critical points of the gray line, which is $\mathrm{KL}(p(Z^n \mid w), p_\pi(Z^n))$. The prior is discrete for finite order $n < \infty$ (Mattingly et al., 2018). Atoms of the prior are maximally different from each other, e.g., for $n = 1$, they are on opposite corners of the parameter space. As the number of samples increases, the separation between atoms of the prior reduces. The prior converges to Jeffreys prior $\pi_J(w) \propto (w(1-w))^{-1}$ as $n \to \infty$.

4.2.2. Second example: reference priors for the classification task

We first discuss a key property of reference priors that enables us to calculate them numerically in high-dimension cases, namely that they are supported on a discrete set in the weight space.

Existence and discreteness of reference priors Rigorous theoretical development of reference priors has been done in the statistics literature. We focus on their applications. We, however, mention some technical conditions under which our development remains meaningful. A reference prior does not exist if $I_\pi(w; Z^n)$ is infinite (Berger et al., 1988). For the concept of a reference prior to remaining meaningful, we make the following technical assumptions. (i) π is supported on a compact set $\Omega \subset \mathbb{R}^p$, and (ii) if $p_\pi(Z^n) = \int_\Omega d(x|z) w \pi(w) p(Z^n \mid w)$ is the marginal, then $\mathrm{KL}(p(Z^n \mid w), p_\pi(Z^n))$ is a continuous function of w for any π. Under these conditions, the n-order prior π_n^* exists and $I_{\pi_n^*}(w; Z^n)$ is finite; see (Zhang, 1994, Lemma 2.14). Now assume that π_n^* exists and is unique up to a set of measure zero. Let $\Omega_n = \{w \in \Omega : \pi_n^*(w) > 0\}$ be the support of π_n^* and Z^n be a discrete random variable with C^n atoms. If $\{p(Z^n \mid w) : w \in \Omega_n\}$ is compact, then π_n^* is discrete with no more than C^n atoms (Zhang, 1994, Lemma 2.18)).

Remark 23 (Blahut-Arimoto algorithm with particles). Since the optimal prior is discrete, we can maximize the mutual information directly by identifying the best set of atoms. We set the prior have the form $\pi_n^* = \sum_{k=1}^{L} L^{-1}\delta(w - w^k)$ where $\{w^1, \dots, w^L\}$ are the L atoms. We call these atoms "particles". Using standard back-propagation, we can then compute the gradient of the objective in (4.3) with respect to each particle (note that each particle's gradient depends upon all other particles).

Let $X^N = \{x_1, ..., x_N\}$ denote the set of the unlabelled data. For each particle w^k in Rem. 23, we compute its corresponding probability distributions $p_{w^k}(Y^N \mid X^N)$, for $Y^N \in \{1, 2, ..., C\}^N$. We compute a principal component analysis (InPCA) of such probabilistic models

$$\{p_{w^1}(\cdot \mid X^N), ..., p_{w^L}(\cdot \mid X^N)\}$$

using a method developed in Quinn et al. (2019a) and visualize the in Fig. 4.2. This experiment demonstrates that we can instantiate reference priors for deep networks in a scalable fashion, even for a large number of particles L. It provides a visual understanding of how atoms of the prior reconstruct diverse tasks based on inputs X^N, just like the atoms in Fig. 4.1.

How to choose the number of atoms L in the reference prior? Each particle in this paper is a deep network, so we must ensure that we maintain a manageable number of atoms in the prior. Abbott and Machta (2019) suggest a scaling law for L in terms of the order of the reference prior n, e.g., $L \sim n^{4/3}$ for a problem with two biased coins. We will instead treat L as a hyper-parameter. This choice is motivated by the emergent low-dimensional structure of the green particles in Fig. 4.2; see the further analysis in in Sec. 4.3.3.

Remark 24 (Variational approximations of reference priors). Nalisnick and Smyth (2017) maximize a lower bound on $I_\pi(w; z)$ and replace the term $p(z) = \int d(x|z)w\pi(w)p(z \mid w)$ in (4.2) by the so-called VR-max estimator $\max_w \log p(z \mid w)$ where the maximum is evaluated across a set of samples from $\pi(w)$ (Li and Turner, 2016). They use a continuous variational family parameterized by neural networks. However, reference priors are supported on a discrete set. Using a continuous

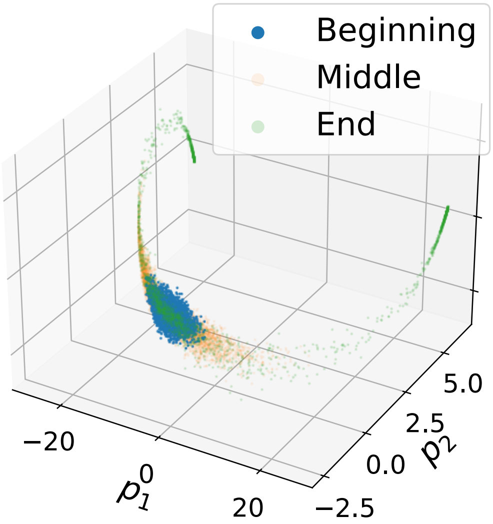


Figure 4.2: **Reference prior (green) for binary classification on MNIST**. A three-dimensional embedding of the probability distributions $p_w(\cdot \mid X^N)$ (reconstructed tasks) of $L = 3000$ atoms in the reference prior after 50,000 iterations of the BA algorithm (green) for a binary classification problem on MNIST (digits 3 vs. 5). Particles were initialized Gaussian (blue) randomly. They are nearby in this embedding because, at initialization, the logits of each particle are uniformly distributed. Orange shows particle locations after 5,000 iterations. As the reference prior objective in (4.3) is optimized, the particles increasingly make more distinguish predictions (orange), and towards the end (green), these probability distributions spread apart boundaries of the hypothesis class.

variational family, e.g., a Gaussian distribution, to approximate π_n^* is computationally beneficial, but it is detrimental to the primary purpose of the prior, namely to discover diverse models. This is also seen in Fig. 4.2, where it would be difficult to construct a variational family whose distributions put mass mostly on the green points. We, therefore, do not use variational approximations.

4.2.3. **Low dimensionality of the space of the tasks**

Why does $n-$reference prior reconstruct the tasks at the boundaries of the hypothesis class? To answer this question, let us recall the definition of the $n-$reference priors (4.3) and notice that,

$$\pi_n^* = \underset{\pi}{\operatorname{argmax}} \int d(x|z) w \pi(w) \mathrm{KL}\left(p(Z^n \mid w),\ p_\pi(Z^n)\right), \tag{4.4}$$

where $p_\pi(Z^n) = \mathbb{E}_{w \sim \pi} p(Z^n \mid w)$ is the average probability distribution on the $n-$reference prior. $n-$reference prior encourages the likelihood $p(Z^n \mid w)$ of atoms in the reference prior over n random samples to be maximally different from the average likelihood $p_\pi(Z^n)$. Fig. 4.3 is a diagram of the hypothesis class $\mathcal{H}$, the red dot nearby the centroid of the hypothesis class represents the average probability distribution $p_\pi(Z^n)$, the blue dots represent the individual probability distributions $p(Z^n \mid w)$ parameterized by the particles in $n-$reference prior. $n-$reference prior maximizes the Kl divergence between the red dot and the blue dots. Therefore, the blue dots are pushed away from the red dot until reaching out the boundaries of the hypothesis class $\mathcal{H}$.

This ability of $n-$reference prior ensures that the probability distributions $p_w(\cdot \mid X^N)$ (tasks) corresponding to the green dots in Fig. 4.2 sketch the outline of the hypothesis class.

Remark 25 (Reference prior depends upon the number of samples and its atoms are diverse models). (4.2) encourages the likelihood $p(Z^n \mid w)$ of atoms in the reference prior to being maximally different from that of other atoms. This gives us intuition as to why the prior should have finite atoms. Consider the covering number in learning theory (Bousquet et al., 2003) where we endow the model space with a metric that measures disagreement between two hypotheses over n samples. Smaller the number of samples n, the smaller the covering number, and the smaller the effective set of models considered. The reference prior is similar. If we only have a few samples n,

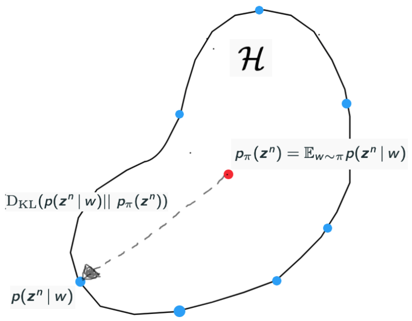

$\mathcal{H}$
$p_\pi(z^n) = \mathbb{E}_{w\sim\pi} p(z^n \mid w)$
$\mathrm{D_{KL}}(p(z^n \mid w) || p_\pi(z^n))$
$p(z^n \mid w)$

Figure 4.3: **This is a diagram of the hypothesis class $\mathcal{H}$.** The red dot nearby the centroid of the hypothesis class represents the average probability distribution $p_\pi(Z^n)$, while the blue dots represent the individual probability distributions $p(Z^n \mid w)$ parameterized by the particles in $n-$reference prior.

then it is not possible for the likelihood in Bayes law to distinguish between a large set of models and assign them different posterior probabilities. The prior, therefore, puts probability mass only on a finite set of atoms, and just like the coin-tossing experiment in Fig. 4.1, these atoms have diverse outputs on the n samples. This ability of the prior to select a small set of representative models is extremely useful for training deep networks with little data.

Emergence of the low dimensionality The neural networks are heavily over-parameterized. In spite of the enormous uncertainty associated with such models, Fig. 4.2 shows that predictions of these models are correspondingly constrained to an effectively low-dimensional hyper-surface bounded with a hierarchy of widths (the green dots in 4.2. Recent research (Mao et al., 2023; Yang et al., 2022; Quinn et al., 2022) advocate that the emergence of the low-dimensional hypothesis class arises from (a) the structure of typical datasets (Goldt et al., 2020; d'Ascoli et al., 2021; Refinetti et al., 2021), e.g., spectral properties, and (b) the fact that typical learning procedures initialize models near a specific region (see the blue dots in 4.5). Along the first direction, recent work on understanding generalization (Yang et al., 2022; Bartlett et al., 2020) has argued that deep networks, as also linear/kernel models, can interpolate without over-fitting if input data have a sloppy spectrum. Work in neuroscience (Simoncelli and Olshausen, 2001; Field, 1994) has also argued for visual data being effectively low-dimensional. Theories in machine learning (Smola and Schölkopf, 1998; Vapnik, 1998) and information theory (Balasubramanian, 1997; Rissanen, 1978) for model selection are based on estimates of the number of models in a hypothesis class that are consistent with the data. The second suspect, namely initialization, suggests that even if the size of the hypothesis class might be very large for deep networks (Dziugaite and Roy, 2017; Bartlett et al., 2017), the subset of the hypothesis space explored by typical learning algorithms might be much smaller.

4.3. Simplicity and Effectiveness: Reconstructed Tasks and their Applications in Small Data Learning

Recent works (Transtrum and Qiu, 2014; Mattingly et al., 2018) have suggested that some human-interpretable effective models are typically obtained by approaching lower-dimensional boundaries of the hypothesis class. In particular, the models at the boundaries are maximally distinguishable from each other and tend to ignore irrelevant features (the multi-parameter examples in (Mattingly et al., 2018)). This ability of simplicity and effectiveness is our primary motivation to explore the applications of the models (reconstructed tasks) at the boundaries of small data learning.

Consider the situation where we are **given limited inputs X^K, their corresponding ground truth labels $\hat{Y}^k$, and a pool of unlabeled inputs X^N**. Building $n-$reference priors require no ground truth labels. Therefore we use unlabelled inputs X^N to build a prior π_n^* and select the models at the boundaries of the hypothesis class. Then we find the solutions that are consistent with the limited labeled data $\left(X^K, \hat{Y}^K\right)$. This learning scheme is named as *semi-supervised learning (SSL)* (Van Engelen and Hoos, 2020). Recall that since π_n^* is a prior, it should not depend on $\left(X^K, \hat{Y}^K\right)$. Just like the construction of the $n-$reference prior in (4.3), we can maximize the mutual information between data predictions and model parameterization

$$\begin{aligned}
I_\pi(Y^n, X^n; w) &= \mathbb{E}_{X^n}\mathbb{E}_{w\sim\pi} \int d(x|z)Y^n p(Y^n \mid X^n, w) \log \frac{p(Y^n \mid X^n, w)}{p_\pi(Y^n \mid X^n)} \\
&= \mathbb{E}_{X^n}\mathbb{E}_{w\sim\pi} \int d(x|z)Y^n p(Y^n \mid X^n, w) \log p(Y^n \mid X^n, w) \\
&\quad - \mathbb{E}_{X^n} \int d(x|z)Y^n p_\pi(Y^n \mid X^n) \log p_\pi(Y^n \mid X^n) \\
&= \mathbb{E}_{X^n}\left[H_\pi(Y^n \mid X^n)\right] - \alpha \mathbb{E}_{X^n}\mathbb{E}_{w\sim\pi}\left[H(Y^n \mid X^n, w)\right],
\end{aligned}$$

where $p_\pi(Y^n \,|\, X^n) = \int d(x|z) w \pi(w) p(Y^n \,|\, X^n, w) = \int d(x|z) w \pi(w) \prod_{i=1}^n p(y_i \,|\, x_i, w)$, X^n is a random subset of X^N with size n ($n << N$), and $\alpha = 1$.

The first step is simply the definition of I_π: it is the KL-divergence between $p(Y^n \,|\, X^n, w)$ and $p_\pi(Y^n \,|\, X^n)$. We assume that inputs X^n and X^N come from the same task. Then we can use samples X^N to compute the expectation over X^n. For the same reason, we also average over outputs Y^n, which are predicted by the network. Let us emphasize that both X^n and Y^n are averaged out in the objective above. Predictions on new samples x are made using the Bayesian posterior predictive distribution

$$p(y \,|\, x, X^K, \hat{Y}^K) \propto \int d(x|z) w \pi_n^*(w) p(y \,|\, x, w) p(\hat{Y}^K \,|\, X^K, w). \tag{4.5}$$

An intuitive understanding of (4.5) Assume for now that we know the number of classes C (although the objective is valid even if that is not the case). If our prior has L particles, then the second term is the average of the per-particle entropy of the predictions. The objective encourages each particle w_i to predict confidently, i.e., to have a small entropy in its output distribution $p(y \,|\, x, w_i)$. The first term is the entropy of the average predictions: $p_\pi(Y^n \,|\, X^n)$, and it is large if particles predict different outputs Y^n for the same inputs X^n, i.e., they disagree with each other. We treat the constant α (which should be 1 in the definition of mutual information) as a hyper-parameter to allow control over this phenomenon. **The reference prior semi-supervised learning objective encourages particles to be dissimilar but confident models (not necessarily correct).**

4.3.1. Practical tricks for implementing reference priors

The reference prior objective is conceptually simple, but it is difficult to implement it directly using deep networks and modern datasets. We next discuss some practical tricks that we have developed.

(1) Order of the reference prior n versus the number of samples Bernardo (1979) set the order of the prior n to be the same as the number of labeled samples. We observe that both do not have to be identical and make a distinction between the two. In our experiments, we restrict the order to $n = 2, 3$. Mathematically, this amounts to computing averages in (4.3) or (4.5) over only sets of n-tuples. This significantly reduces the class of models considered in the reference prior by *pretending* that there is a small number of labeled samples available for training the task—which is useful, and also true in practice, for over-parametrized deep networks. This choice is also motivated by the low-dimensional structure in the reference prior in Fig. 4.2. Note that we are *not* restricting to small order n for computational reasons, i.e., computing the expectation over all classes Y^n in (4.5) can be done in a single forward pass.

(2) Using cross-entropy loss to bias particles towards good parts of the weight space The posterior (4.5) suggests that we should first compute the prior and then weight each particle by the likelihood of the labeled data. In practice, we combine these two steps into a single objective

$$\max_{\pi} \ \gamma I_\pi(w; Y^n, X^n) + \mathbb{E}_{w\sim\pi}\left[\log p(\hat{Y}^K \mid X^K, w)\right], \tag{4.6}$$

where γ is a hyper parameter, $X^K, \hat{Y}^K$ are labeled samples. (4.6) allows us to directly obtain particles that both have high probability under the prior and a high likelihood. This is different from the correct Bayesian posterior (which would set $\gamma = 1$, we use $\gamma = 1/2$) but it is a trick often employed in the Bayesian inference literature. The second term restricts the search space for the particles in $\pi(w)$.

(3) Data augmentation State-of-the-art SSL methods use heavy data augmentation, e.g., RandAugment (Cubuk et al., 2020) and CTAugment (Berthelot et al., 2019a), which both have about 20 transformations. Some are weak augmentations, such as mirror flips and crops, while others are strong augmentations, like color jitter. Methods such as FixMatch (Sohn et al., 2020) or MixMatch (Berthelot et al., 2019b) use weak augmentations to get soft labels for predictions on strong augmentations.

We compute the entropy term $H(Y^n \mid X^n, w)$ in (4.5) using the distribution

$$p_G(y \mid x, w) = \mathbb{E}_{g\sim G}[p_w(y \mid g(x), w)]$$

where $G = G_1 \cup G_2$ is the set of weak (G_1) and strong (G_2) augmentations. Let $g_i \sim G_i$ be an augmentation and denote $p_{g_i} \equiv p(y \mid g_i(x), w)$ for $i \in \{1, 2\}$. In every mini-batch we use $p_G(y \mid x, w) \approx \tau p_{g_1} + (1-\tau) p_{g_2}$ where τ is a hyper-parameter. This gives accuracy that is reasonable (about 87% for 500 samples) but a bit lower than state-of-the-art SSL methods. We noticed that if we use an upper bound on the entropy from Jensen's inequality

$$-\mathbb{E}_{X^n} \int d(x|z) Y^n p_G(Y^n \mid X^n, w) \left[\tau \log p_{g_1} + (1-\tau) \log p_{g_2}\right] \tag{4.7}$$

Then we can close this gap in accuracy (see Table 4.1). This is perhaps because the cross-entropy terms, e.g., $-p_{g_1} \log p_{g_2}$, force the predictions of the particles to be consistent across both types of augmentations, just like the objective in FixMatch or MixMatch. Our formulation is thus useful to not only understand SSL but also to tweak it to perform as well as current methods and thereby shed light on the theoretical underpinnings of their performance.

(4) Computing $H(Y^n \mid X^n, w)$ A number of SSL methods work by creating pseudo labels from weakly augmented data, which seems to be a key ingredient of good accuracy in our experience with these methods. We tried two heuristics to compute the entropy term $H(Y^n \mid X^n, w)$ that are motivated by these papers. First, we follow FixMatch and only use unlabeled data with confident predictions to compute $H(Y^n \mid X^n, w)$. A datum x contributes to the objective only if $\max_y p(y|g_1(x), w) > 0.95$. Changing this threshold does not lead to deterioration of the accuracy as we see in Table 4.6, so this heuristic need not be used while building the reference prior. Second, if G_1 is the set of weak augmentations (see previous point), methods like FixMatch and MixMatch use $\text{argmax}_y\, p(y \mid g_1(x), w)$ as a pseudo-label but do not update this using the back-propagation gradient. This prevents the more reliable predictions on G_1 from changing. As a result, the entropy term $-\tau^2 p_{g_1} \log p_{g_1}$ is a constant in (4.7). To normalize the terms coming from τ in (4.7), we set γ in (4.6) to $1/(1-\tau^2)$ instead of 1. We have also developed an argument to choose the appropriate value of $\tau = 1/3$ that we explain in Sec. 4.6.1. This second heuristic seems essential, in Table 4.6, we obtain only 10% accuracy without this heuristic.

4.3.2. **Empirical study**

We evaluate on CIFAR-10 and CIFAR-100 (Krizhevsky, 2009). We use 50–1000 labeled samples, i.e., 5–100 samples/class and use the rest of the samples in the training set as unlabeled samples. All experiments use the WRN 28-2 architecture (Zagoruyko and Komodakis, 2016), same as in Berthelot et al. (2019b).

For all our experiments, the reference prior is of order $n = 2$ and has $L = 4$ particles. We run all our methods for 200 epochs, with $\tau = 1/3$ in (4.7) and $\alpha = 0.1$ in (4.5). We set $\gamma = (1 - \tau^2)^{-1}$ as discussed in Sec. 4.3.1. For inference, each particle maintains an exponential moving average (EMA) of the weights (this is common in SSL (Tarvainen and Valpola, 2017)). Sec. 4.6.1 provides more details.

Baselines We compare to a number of recent methods such as FixMatch (Sohn et al., 2020), MixMatch (Berthelot et al., 2019b), DASH (Xu et al., 2021), SelfMatch (Kim et al., 2021), Mean Teacher (Tarvainen and Valpola, 2017), Virtual Adversarial Training (Miyato et al., 2018), and Mixup (Berthelot et al., 2019b).

Table 4.1 compares the accuracy of different SSL methods on CIFAR-10. We find that the reference prior approach is competitive with a number of existing methods, e.g., it is remarkably close to FixMatch on all sample sizes (notice the error bars). There is a gap in accuracy at small sample sizes (40–50) when compared to recent methods. It is important to note that these recent methods employ a number of additional tricks, e.g., FlexMatch implements curriculum learning on top of FixMatch, DASH and FlexMatch use different thresholding for weak augmentations (this increases their accuracy by 2-5%), SelfMatch has higher accuracies because of a self-supervised pre-training stage, FixMatch (CTA) outperforms its RA variant by 1.5% which indicates CTA augmentation is beneficial (we used RA). It is also extremely expensive to train SSL algorithms for 1000 epochs (all methods in Table 4.1 do so), we trained for 200 epochs.

This experiment shows that our approach to small data learning such as SSL can obtain results that are competitive to sophisticated empirical methods without being explicitly formulated to enforce

Method	Samples				
	50	100	250	500	1000
Mixup	-	-	52.57	63.86	74.28
VAT	-	-	63.97	73.89	81.32
Mean Teacher	-	-	52.68	57.99	82.68
MixMatch	64.21*	80.29*	88.91*	90.35*	92.25*
FixMatch (RA)	86.19 ± 3.37 (40)	90.12*	94.93 ± 0.65	93.91*	94.3*
FixMatch (CTA)	88.61 ± 3.35 (40)	-	94.93 ± 0.33	-	-
DASH (RA)	86.78 ± 3.75 (40)	-	95.44 ± 0.13	-	-
DASH (CTA)	90.84 ± 4.31 (40)	-	95.22 ± 0.12	-	-
SelfMatch	93.19 ± 1.08 (40)	-	95.13 ± 0.26	-	-
FlexMatch	95.03 ± 0.06 (40)	-	95.02 ± 0.09	-	-
Deep Reference Prior	85.45 ± 2.12	88.53 ± 0.67	92.13 ± 0.39	92.94 ± 0.22	93.48 ± 0.24

Table 4.1: **Classification accuracy of different semi-supervised learning methods on CIFAR-10. Note:** RA and CTA in the methods column indicate that RandAugment or CTAugment were used for augmentations. Entries with * were evaluated by us using open-source implementations from the original authors for 256 epochs. All other entries are from original papers. Entries with "(40)" indicate that 40 labeled samples were used instead of 50.

properties like label consistency with respect to augmentations. This also indicates that reference priors could be a good way to explain the performance of these existing methods, which is one of our goals in this paper.

4.3.3. Ablation and analysis

This section presents ablation and analysis experiments for SSL on CIFAR-10 with 1000 labeled samples. We study the reference prior for different settings (i) varying the order n of the prior, (ii) varying the number of particles in the BA algorithm (L), (iii) exponential moving averaging of the weights for each particle. We also study the two entropy terms in the reference prior objective individually.

We use a reference prior of order $n = 2$ in all our experiments. We see in Table 4.2 that **changing the order of the prior** leads to marginal (about 1%) changes in the accuracy.

We next **vary the number of particles** in the prior in Table 4.3 and find that the accuracy is relatively consistent when the number of particles varies from $L = 2$ to $L = 16$. This seems surprising

Method	Order (→)	2	3	4	5
Deep Reference Prior ($K = 4$)		91.76	90.53	91.51	91.36

Table 4.2: The order of the reference prior has a minimal impact on the accuracy.

Method	#Particles (→)	2	4	8	16
Deep Reference Prior ($n = 2$)		91.3	91.76	89.79	90.72

Table 4.3: Number of particles has a minimal impact on accuracy.

because a reference prior ideally should have an infinite number of atoms, when it approximates Jeffreys prior. Our experiment in Fig. 4.2 provides insight into this phenomenon. It shows that the manifold of diverse predictions is low-dimensional. Particles of the reference prior only need to span these few dimension and we can fruitfully implement our approach using very few particles.

Effect of exponential moving averaging (EMA) We use EMA on the weights of each particle (independently). Table 4.4 analyzes the impact of EMA. As noticed in other semi-supervised learning works (Berthelot et al., 2019b; Sohn et al., 2020), EMA improves the accuracy by 2-3% regardless of the number of labeled samples used.

Method	#Samples (→)	50	100	250	500	1000
EMA		85.45 ± 2.12	88.53 ± 0.67	92.13 ± 0.39	92.94 ± 0.22	93.48 ± 0.24
No EMA		82.36 ± 2.13	85.64 ± 0.43	89.75 ± 0.36	90.06 ± 1.71	91.57 ± 0.25

Table 4.4: Using EMA for weights of each particle is beneficial and improves accuracy by 2-3%.

The two entropy terms in the reference prior objective Fig. 4.4 (left) shows how, because of the entropy term $H_\pi(Y^n \mid X^n)$, the accuracy of particles is quite different during training. Particles have different predictive abilities (7% range in test error) but the Bayesian posterior predictive distribution has a higher accuracy than any of them. Fig. 4.4 (right) tracks the two entropy terms in the objective. For large number of labeled data (500, blue) the entropy $H_\pi(Y^n \mid X^n)$ which should always be higher than $H(Y^n \mid X^n, w)$ in (4.5) is lower (this is not the case for 50 samples, red). This is likely a result of the cross-entropy term in the modified objective in (4.6) which narrows the search space of the particles. This experiment is an important insight into the working of existing semi-supervised learning methods as well, all of which also have a similar cross-entropy objective

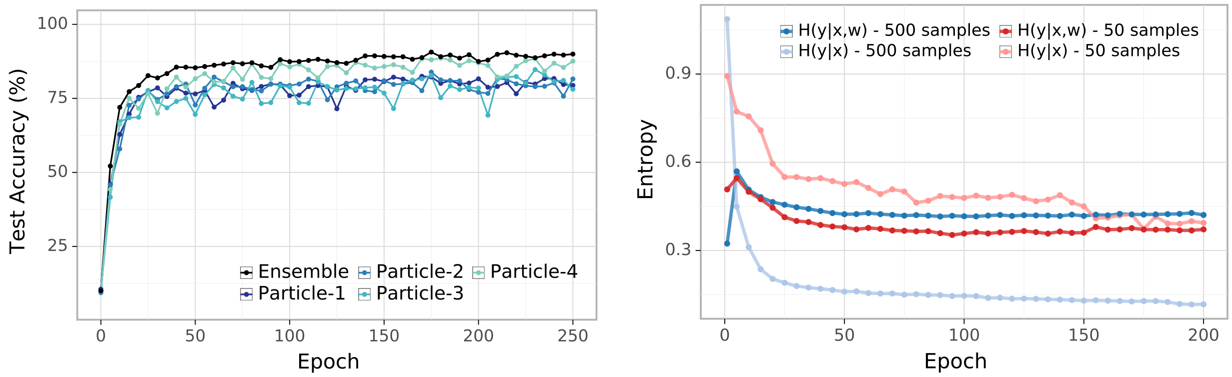


Figure 4.4: **(Left)** Accuracy of individual particles in the prior during training (250 labeled samples). The individual particles have diverse predictions due to the entropy term $H_\pi(Y^n \mid X^n)$, the accuracy of the ensemble is larger than the accuracy of any single particle. **(Right)** Evolution of entropy terms $H(Y^n \mid X^n, w)$ and $H_\pi(Y^n \mid X^n)$ for two cases (500 labeled samples and 50 labeled samples). While $H_\pi(Y^n \mid X^n)$ is expected to be larger than $H(Y^n \mid X^n, w)$ in (4.5) since KL-divergence is non-negative, this is not always the case since we approximate $H(Y^n \mid X^n, w)$ by an upper-bound obtained from Jensen's inequality for data augmentation as discussed in Sec. 4.3.1.

in their formulation. It points to the fact that at large sample-sizes, the cross-entropy loss and not the semi-supervised learning objective could dominate the training procedure.

4.4. Reference priors for a Two-Stage Experiment

We first develop the idea using generic random variables Z^n. Consider a situation when we **see data in two stages, first Z^m, and then Z^n.** How should we select a prior, and thereby the posterior of the first stage, such that the posterior of the second stage makes maximal use of the new n samples? We can extend the idea in (4.3) in a natural way to address this question. We can **maximize the KL-divergence between the posterior of the second stage and the posterior after the first stage, on average, over samples Z^n.**

Since we have access to samples Z^m, we need not average over them, we can compute the posterior $p(w \mid Z^m)$ from these samples given the prior $\pi(w)$. First, notice that $p(w, Z^n \mid Z^m) =$

$p(w \,|\, Z^{m+n}) p(Z^n \,|\, Z^m) = p(Z^n \,|\, w) p(w \,|\, Z^m)$. We can now write

$$\begin{aligned}
\pi^*_{n\,|\,m} &= \underset{\pi}{\operatorname{argmax}}\, I_{p(w\,|\,Z^m)}(w; Z^n) \\
&:= \int d(x|z) Z^n p(Z^n \,|\, Z^m)\ \mathrm{KL}(p(w \,|\, Z^{m+n}), p(w \,|\, Z^m)) \\
&= \int d(x|z) w p(w \,|\, Z^m) \int d(x|z) Z^n p(Z^n \,|\, w) \log \frac{p(Z^n \,|\, w)}{p(Z^n \,|\, Z^m)},
\end{aligned} \tag{4.8}$$

where $p(w \,|\, Z^m) \propto p(Z^m \,|\, w)\pi(w)$ and $p(Z^n \,|\, Z^m) = \int d(x|z) w p(Z^n \,|\, w) p(w \,|\, Z^m)$. The key observation is that if the reference prior (4.3) has a unique solution, we should have that the optimal $p(w \,|\, Z^m) \equiv \pi^*_n(w)$. This leads to

$$\pi^*_{n\,|\,m}(w) \propto \pi^*_n(w)\ p(Z^m \,|\, w)^{-1}. \tag{4.9}$$

This prior puts *less* probability on regions which have high likelihood on old data Z^m whereby the posterior is maximally informed by the new samples Z^n. Given knowledge of old data, the prior *downweighs regions* in the weight space that could bias the posterior of the new data. We also have $\pi^*_{n\,|\,m} = \pi^*_n$ for $m = 0$ which is consistent with (4.3). As $m \to \infty$, this prior ignores the part of the weight space that was ideal for Z^m. See Sec. 4.6.5 for an example.

Remark 26 (Averaging over Z^m in the two-stage experiment). If we do not know the outcomes Z^m yet, the prior should be calculated by averaging over both Z^m, Z^n

$$\begin{aligned}
\pi^* &= \underset{\pi}{\operatorname{argmax}} \int d(x|z) Z^m p(Z^m) I_{p(w\,|\,Z^m)}(w; Z^n) \\
&:= I_\pi(w; Z^{m+n}) - I_\pi(w; Z^m) \\
&= H(w \,|\, Z^m) - H(w \,|\, Z^{m+n}).
\end{aligned} \tag{4.10}$$

The encourages multiple explanations of initial data Z^m, i.e., high $H(w \,|\, Z^m)$, so as to let the future samples Z^n select the best one among these explanations, i.e., reduce the entropy $H(w \,|\, Z^{m+n})$. It is interesting to note that neither is this two-stage prior equivalent to maximizing $I_\pi(w; Z^{m+n})$, nor is it simply the optimal prior corresponding to objectives $I_\pi(w; Z^m)$ or $I_\pi(w; Z^n)$. Both (4.9) and (4.10) therefore indicate that two-stage priors are useful when we have *some* data *a priori*, this can be either unlabeled samples from the same task, or labeled samples from some other task.

Remark 27 (A softer version of the two-stage reference prior). The objective in (4.10) resembles the predictive information bottleneck (IB) of Bialek et al. (2001), or its variational version in Alemi (2020), which seek to learn a representation, say w, that maximally forgets past data while remaining predictive of future data

$$\max_{p(w \mid Z^m)} I(w; Z^n) - \beta I(w; Z^m). \tag{4.11}$$

The parameter β in (4.11) gives this objective control over how much information from the past is retained in w. We take inspiration from this and construct a variant of (4.9)

$$\begin{aligned} \pi^{\beta}_{n \mid m}(w) &\propto \pi^*_n(w) p(Z^m \mid w)^{-\beta} \quad \text{for } \beta \in (0,1). \\ \Rightarrow p(w \mid Z^{m+n}) &\propto p(Z^n \mid w) p(Z^m \mid w)^{1-\beta} \pi^*_n(w). \end{aligned} \tag{4.12}$$

We should use $\beta = 0$ when we expect that data from the first stage Z^m is similar to data Z^n from the second stage. This allows the posterior to *benefit* from past samples. If we expect that the data are different, then $\beta = 1$ ignores regions in the weight space that predict well for Z^m. This is similar to the predictive IB where a small β encourages remembering the past and $\beta = 1$ encourages forgetting.

4.4.1. Reference priors for transfer learning

Consider the two-stage experiment where in the first stage we obtain m samples (X^m_s, Y^m_s) from a "source" task P^s and the second stage consists of n samples (X^n_t, Y^n_t) from the "target" task P^t. Our goal is to calculate a prior $\pi(w)$ that best utilizes the target task data.

Bayesian inference for this problem involves first computing the posterior

$$p(w \mid X^m_s, Y^m_s) \propto p(Y^m_s \mid w, X^m_s) \pi(w)$$

from the source task and then using it as a prior to compute the posterior for the target task $p(w \mid X_t^n, Y_t^n, X_s^m, Y_s^m)$. Just like Sec. 4.1.1, **the key idea again is to maximize the KL-divergence between the two posteriors** $\mathrm{KL}\left(p(w \mid X_t^n, Y_t^n, X_s^m, Y_s^m),\ p(w \mid X_s^m, Y_s^m)\right)$, but averaged over samples X_s^m and X_t^n.

Case 1: Access to unlabeled data from the source X_s^m and the target task X_t^n We should average the KL-divergence over both the source and target predictions Y_s^m and Y_t^n and maximize

$$\mathbb{E}_{X_s^m,\, X_t^n,\, Y_s^m \mid X_s^m,\, Y_t^n \mid X_t^n} \left[\mathrm{KL}\left(p(w \mid X_t^n, Y_t^n, X_s^m, Y_s^m), p(w \mid X_s^m, Y_s^m)\right)\right] \tag{4.13}$$

over the prior π. Here $p_\pi(Y_s^m \mid X_s^m) = \mathbb{E}_{w \sim \pi} p(Y_s^m \mid X_s^m, w)$ and $p_\pi(Y_t^n \mid X_t^n) = \mathbb{E}_{w \sim \pi} p(Y_t^n \mid X_t^n, w)$, respectively. Note that averages over X_s^m and X_t^n are computed using samples while averages over $Y_s^m \mid X_s^m$ and $Y_t^n \mid X_t^n$ are computed using the model's predictions.

Case 2: X_s^m, Y_s^m are fixed and known, and we have a pool of unlabeled target data X_t^n Since we already know the labels for the source task, we will only average over X_t^n and Y_t^n and maximize

$$\mathbb{E}_{X_t^n, Y_t^n \mid X_t^n} \mathrm{KL}\left(p(w \mid X_t^n, Y_t^n, X_s^m, Y_s^m), p(w \mid X_s^m, Y_s^m)\right); \tag{4.14}$$

here $p_\pi(Y_t^n \mid X_t^n) = \int d(x|z) w \pi(w) p(Y_t^n \mid X_t^n, w)$.

Remark 28 (Connecting (4.13) and (4.14) to practice). Both objectives can be written down as

$$\pi^* = \underset{\pi}{\mathrm{argmax}}\, I_\pi(w; Y_t^n, X_t^n, X_s^m, Y_s^m) - I_\pi(w; X_s^m, Y_s^m) \tag{4.15}$$

with the distinction that while in Case 1, we average over all quantities, namely $p(X_s^m)$, $p(Y_s^m)$, $p(X_t^n)$, $p(Y_t^n)$ while in Case 2, we fix X_s^m and Y_s^m to the provided data from the source task. Case 2 is what is typically called transfer learning. Case 1, where one has access to *only unlabeled data* from a source task *that is different from the target task* is not typically studied in practice. Like (4.12), we can again introduce a coefficient β on the second term in (4.15) to handle the relatedness between source and target tasks.

4.4.2. Empirical study

We evaluate on CIFAR-100 (Krizhevsky, 2009). We construct 20 five-way classification tasks from CIFAR-100 and use 1000 labeled samples from the source and 100 labeled samples from the target task. All experiments use the WRN 28-2 architecture (Zagoruyko and Komodakis, 2016), same as in Berthelot et al. (2019b).

For all our experiments, the reference prior is of order $n = 2$ and has $L = 4$ particles. We run all our methods for 200 epochs, with $\tau = 1/3$ in (4.7) and $\alpha = 0.1$ in (4.5). We set $\gamma = (1 - \tau^2)^{-1}$ as discussed in Sec. 4.3.1. For inference, each particle maintains an exponential moving average (EMA) of the weights (this is common in SSL (Tarvainen and Valpola, 2017)). Sec. 4.6.1 provides more details. Just like we did in Sec. 4.3.1, we instantiate (4.12) and (4.14), by combining prior selection, pre-training on the source task and likelihood of the target task, into one objective,

$$\gamma I_\pi(w; Y_t^n, X_t^n) + \mathbb{E}_{w\sim\pi}\left[\log p(w, Y_t^n \mid X_t^n)\right] + (1-\beta)\mathbb{E}_{w\sim\pi}\left[\log p(w, Y_s^m \mid X_s^m)\right], \tag{4.16}$$

where $\gamma = 1/2$ and $\beta = 1/2$ are hyper-parameters, (X_s^m, Y_s^m) are labeled data from the source task ($m = 1000$), (X_t^n, Y_t^n) are labeled data from the target task ($n = 100$) and X_t^n are unlabeled samples from the target task (all other samples).

Baselines We use three methods: (a) fine-tuning, which is a very effective strategy for transfer learning (Dhillon et al., 2020; Kolesnikov et al., 2020) but it cannot use unlabeled target data, (b) using only labeled target data (this is standard supervised learning), and (c) using only labeled and unlabeled target data without any source data (this is $\beta = 1$ in (4.16)). Fig. 4.5 compares the performance for pairwise transfer across 5 tasks from CIFAR-100. Our reference prior objective in (4.16) obtains much better accuracy than fine-tuning which indicates that it leverages the unlabeled target data effectively. For each task, the accuracy is much better than both standard supervised learning and semi-supervised learning using our own reference prior approach (4.6); both of these indicate that the labeled source data is being used effectively in (4.16).

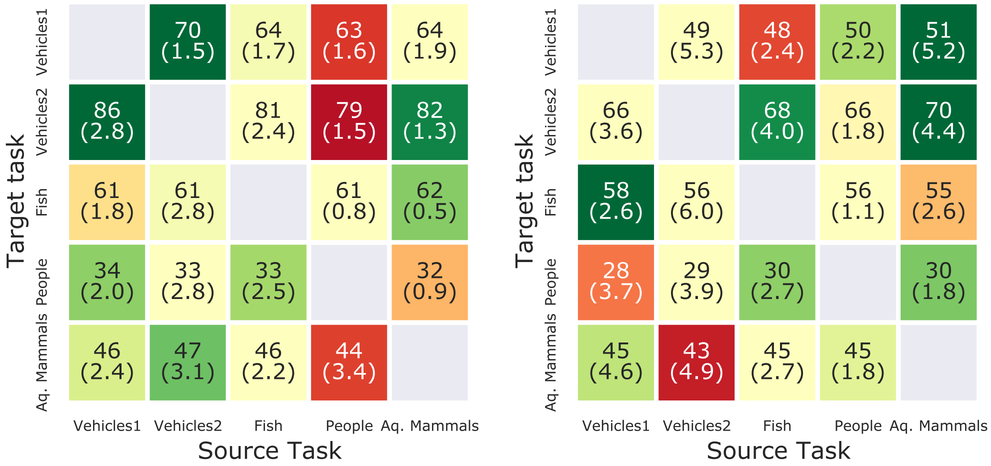


Figure 4.5: **Top: Accuracy (%) of deep reference priors (left) and fine-tuning (right) for transfer learning tasks in CIFAR-100**. Cells are colored red/green relative to the median accuracy of each row. Darker shades of green indicate that the source task is more suitable for transfer. For example, Vehicles-1 as source is the best for all tasks according to the reference prior (left) (which is optimal in theory) but fine-tuning cannot replicate this. The accuracy of cells in the left panel is better than the corresponding cells on the right, e.g., the gap in accuracy is 34.8% for Vehicles 2 → Vehicles 1. **Bottom: Accuracy (%) of supervised learning and SSL for all 5 tasks**. Each number here should be compared to the corresponding row of the matrices in the top panel, e.g., Vehicles 2 has 86% accuracy when transferred from Vehicles 1 using our transfer method (left), it has 66% accuracy from fine-tuning (right), while the same task achieves 63.2% accuracy when trained by itself using supervised learning (table first row) and 75.2% accuracy when trained using unlabeled target data (table second row). Therefore the reference prior-based transfer objective can leverage both labeled source data as well as unlabeled target data. This pattern is consistent for all tasks.

4.5. Related Work and Discussion

Reference priors in Bayesian statistics We build upon the theory of reference priors which was developed in the objective Bayesian statistics literature Bernardo (1979); Berger et al. (1988, 2009). The main idea used in our work is that non-asymptotic reference priors allow us to exploit the finite samples from the task in a fundamentally different way than classical Bayesian inference. If the number of samples from the task available to the learner is finite, then the prior should also select only a finite number of models. Reference priors are not common in the machine learning literature. A notable exception is Nalisnick and Smyth (2017) who optimize a variational lower bound and demonstrate results on small-scale problems. The main technical distinction of our work is that we explicitly use the discrete prior instead of a variational approximation.

Information theory Discreteness is seen in many problems with an information-theoretic formulation, e.g., capacity of a Gaussian channel under an amplitude constraint (Smith, 1971), neural representations in the brain Laughlin (1981), and biological systems (Mayer et al., 2015). (Mattingly et al., 2018; Abbott and Machta, 2019) have developed these ideas to study how reference priors select "simple models" which lie on certain low-dimensional "edges" of the model space. We believe that the methods developed in our paper are effective *because* of this phenomenon. Our choice of using a small order n for the prior is directly motivated by their examples.

Semi-supervised learning Our formulation sheds light on the working of current SSL methods. For example, the reference prior can automatically enforce consistency regularization of predictions across augmentations (Tarvainen and Valpola, 2017; Berthelot et al., 2019b), as we discuss in Sec. 4.3.1. Similarly, minimizing the entropy of predictions on unlabeled data, either explicitly (Grandvalet et al., 2005; Miyato et al., 2018) or using pseudo-labeling methods (Lee et al., 2013; Sajjadi et al., 2016), is another popular technique. This is automatically achieved by the objective in (4.5). Disagreement-based methods (Zhou and Li, 2010) employ multiple models and use confident models to soft-annotate unlabeled samples for others. Disagreements in our formulation are encouraged by the entropy $H_\pi(Y^n \mid X^n)$ in (4.5). If $p_\pi(Y^n \mid X^n)$ is uniform, which is encouraged by the reference prior objective, particles disagree strongly with each other.

Transfer learning is a key component of a large number of applications today, e.g, (Devlin et al., 2019; Kolesnikov et al., 2020) but a central question that remains unanswered is how one should pretrain a model if the eventual goal is to transfer to a target task. There have been some attempts at addressing this via the Information Bottleneck, e.g., Gao and Chaudhari (2020a). This question becomes particularly challenging when transferring across domains, or for small sample sizes (Davatzikos, 2019). Reference priors are uniquely suited to tackle this question: our two-stage experiment in Sec. 4.4 is the *optimal* way pretain on the source task. As our experiments show, this is better than fine-tuning in the low-sample regime Sec. 4.4.2.

4.6. Appendix

4.6.1. Details of the experimental setup

Architecture For experiments on CIFAR-10 (Sec. 4.3.2) and CIFAR-100 (Sec. 4.4.2), we consider a modified version of the Wide-Resnet 28-2 architecture (Zagoruyko and Komodakis, 2016), which is identical to the one used in Berthelot et al. (2019b). This architecture differs from the standard Wide-Resnet architecture in a few important aspects. The modified architecture has Leaky-ReLU with slope 0.1 (as opposed to ReLU), no activations or batch normalization before any layer with a residual connection, and a momentum of 0.001 for batch-normalization running mean and standard-deviation (as opposed to 0.1, in other words these statistics are made to change very slowly). We observed that the change to batch-normalization momentum has a very large effect on the accuracy of semi-supervised learning.

For experiments on MNIST (Sec. 4.6.3), we use a fully-connected network with 1 hidden layer of size 32. We use the hardtanh activation in place of ReLU for this experiment; this is because maximizing the mutual information has the effect of increasing the magnitude of the activations for ReLU networks. One may use weight decay to control the scale of the weights and thereby that of the activations but in an effort to implement the reference prior exactly, we did not use weight decay in this model. Note that the nonlinearities for the CIFAR models are ReLUs.

Datasets For semi-supervised learning, we consider the CIFAR-10 dataset with the number of labeled samples varying from 50–1000 (i.e., 5–100 labeled samples per class). Semi-supervised learning experiments use all samples that are not a part in the labeled set, as unlabeled samples.

For transfer learning, we construct two tasks from MNIST (task one is a 5-way classification task for digits 0–4, and task two is another 5-way classification task for digits 5–9). For this experiment, we use labeled source data but do not use any labeled target data. This makes our approach using a reference prior similar to a purely unsupervised method.

The CIFAR-100 dataset is also utilized in the transfer learning setup (Sec. 4.4.2). We consider five 5-way classification tasks from CIFAR-100 constructed using the super-classes. The five tasks considered are Vehicles-1, Vehicles-2, Fish, People and Aquatic Mammals. The selection of these tasks were motivated from the fact that some pairs of tasks are known to positively impact each other (Vehicles-1, Vehicles-2), while other pairs are known to be detrimental to each other (Vehicles-2, People); see the experiments in Ramesh and Chaudhari (2022b).

Optimization SGD with Nesterov momentum on a Cosine-annealed learning rate schedule with warmup was used in our experiments on CIFAR-10 and CIFAR-100. The initial learning rate was set to $0.03 \times K$ where K denotes the number of particles. The scaling factor of K exists to counteract the normalization constant in the objective from averaging across all particles. The momentum coefficient for SGD was set to 0.9 and weight decay to $5K^{-1} \times 10^{-4}$. Mixed-precision (32-bit weights, 16-bit gradients) was used to expedite training. Training was performed for 200 epochs unless specified otherwise.

Experiments on MNIST also used SGD for computing the reference prior. SGD was used with a constant learning rate of 0.001 with Nesterov's acceleration, momentum coefficient of 0.9 and weight decay of 10^{-5}.

Definition of a single Epoch Note that since we iterate over the unlabeled and labeled data (each with different number of samples), the notion of what is an epoch needs to be defined differently. In our work, one epoch refers to 1024 weight updates, where each weight update is calculated using a batch-size of 64 for the labeled data of batch size 64, and a batch-size of 448 for the unlabeled data.

Exponential Moving Average (EMA) In all CIFAR-10 and CIFAR-100 experiments, we also implement the Exponential Moving Average (EMA) (Tarvainen and Valpola, 2017). In each step, the EMA model is updated such that the new weights are the weighted average of the old EMA model weights, and the latest trained model weights. The weights for averaging used in our work (and most other methods) are 0.999 and 0.001 respectively. Note that EMA only affects the particle when it is used for testing, it does not affect how weight updates are calculated during training. We exclude batch-normalization running mean and variance estimates in EMA.

Data Augmentations We use random-horizontal flips and random-pad-crop (padding of 4 pixels on each side) as weak augmentations for the CIFAR-10 and CIFAR-100 datasets. For SSL experiments on CIFAR-10, we use RandAugment (Cubuk et al., 2020) for strong augmentations. No data augmentations were used for MNIST.

Picking the value of τ in (4.7) Let G_1 and G_2 be the sets of weak and strong augmentations respectively. For $g_1 \sim G_1$ and $g_2 \sim G_2$, let us write down the upper bound in (4.7) from Jensen's inequality in detail

$$\mathbb{E}_{X^n} \int d(x|z) Y^n \left[-\tau^2 p_{g_1} \log p_{g_1} - \tau(1-\tau) p_{g_2} \log p_{g_1} - (1-\tau)\tau p_{g_1} \log p_{g_2} - (1-\tau)^2 p_{g_2} \log p_{g_2} \right].$$

The upper bound is thus a weighted sum of the entropy terms $-p_{g_1} \log p_{g_1}, -p_{g_2} \log p_{g_2}$, and cross entropy terms $-p_{g_2} \log p_{g_1}, -p_{g_1} \log p_{g_2}$. If we were to pick $\tau = 1/2$ like FixMatch, then since $(1-\tau)^2 + \tau^2 = 2\tau(1-\tau)$ for $\tau = 1/2$, the entropy and cross entropy terms will contribute equally to the loss function. However in practice, since we do not update p_{g_1} using the back-propagation gradient to protect the predictions from deteriorating on the weakly augmented images, one of the entropy terms $-p_{g_1} \log p_{g_1}$ is dropped. In such a situation, to ensure that cross entropy and entropy terms provide an equal contribution to the gradient, we would like $(1-\tau)^2 = 2\tau(1-\tau)$ which gives $\tau = 1/3$.

4.6.2. **Overview of the implementation**

We provide an overview of the implementation of deep reference priors.
For more details see https://github.com/rahul13ramesh/deep_reference_priors.

Let a mini-batch from the labeled dataset be denoted by $\{(x_i, y_i)\}_{i=1}^{b}$ and a mini-batch from the unlabeled dataset be denoted by $\{(x^u_{i0}, x^u_{i1}, \cdots, x^u_{in})\}_{i=1}^{b_u}$ where n is the order of the reference prior. Note the distinction in the two mini-batches, i.e. the unlabeled mini-batch consists of a set of n-tuples unlike the labeled mini-batch. Let g_1 and g_2 be functions that perform weak and strong augmentations respectively. The reference prior objective is used to train K particles $\{p_{w_k}\}_{k=1}^{K}$.

For the sample X^n, we compute $p(Y^n \,|\, X^n, w_k)$ as follows:

$$p(Y^n \,|\, X^n, w_k) = \tau p(Y^n \,|\, g_1(X^n), w_k) + (1-\tau) p(Y^n \,|\, g_2(X^n), w_k)$$

The reference prior loss ,requires us to compute the terms

$$\begin{aligned}
\mathbb{E}_{w\sim\pi}\left[H(Y_i^n \,|\, X_i^n, w)\right] &= \sum_{k=1}^{K} \pi(w_k) \sum_{y\in\mathcal{Y}^n} \left(-p(y \,|\, X_i^n, w_k) \log(p(y \,|\, X_i^n, w_k))\right) \\
&= \sum_{k=1}^{K} \pi(w_k) \sum_{y\in\mathcal{Y}^n} \left(-\prod_{j=1}^{n} p(y \,|\, x_{ij}^u, w_k)\right) \log\left(\prod_{j=1}^{n} p(y \,|\, x_{ij}^u, w_k)\right) \\
&= \sum_{k=1}^{K} \pi(w_k) \sum_{j=1}^{n} \sum_{y\in\mathcal{Y}} -p(y \,|\, x_{ij}^u, w_k) \log\left(p(y \,|\, x_{ij}^u, w_k)\right) \\
&\le \sum_{k=1}^{K} \pi(w_k) \sum_{j=1}^{n} \sum_{y\in\mathcal{Y}} -p(y \,|\, x_{ij}^u, w_k) \left[\tau \log p(y \,|\, g_1(x_{ij}^u), w_k) + (1-\tau) \log p(y \,|\, g_2(x_{ij}^u), w_k)\right],
\end{aligned}$$

and

$$\begin{aligned}
H(Y_i^n \,|\, X_i^n) &= \sum_{y^n\in\mathcal{Y}^n} -p(y^n \,|\, x_i^u) \log(p(y^n \,|\, x_i^u)) \\
&= \sum_{y^n\in\mathcal{Y}^n} -\left(\sum_{k=1}^{K} \pi(w_k) p(y^n \,|\, x_i^u, w_k)\right) \log\left(\sum_{k=1}^{K} \pi(w_k) p(y^n \,|\, x_i^u, w_k)\right).
\end{aligned}$$

In our implementation, we set $\pi(w_k) = \frac{1}{K}$. We observed no improvement in accuracy if the elements of π were trainable weights.

Input data consists of a mini-batch of labeled data $\{(x_i, y_i)\}_{i=1}^{b}$ and unlabeled data $\{x_{i0}^u, x_{i1}^u, \cdots, x_{in}^u)\}_{i=1}^{b_u}$ and a user-determined order n.

Trainable weights are the weights of the K neural networks (also called particles) $\{p_{w_k}\}_{k=1}^{K}$.

Define

$$f(x, y, w) = \tau p_w(y \,|\, g_1(x)) + (1-\tau) p_w(y \,|\, g_2(x)),$$

$$f_{\log}(x, y, w) = \tau \log p_w(y \,|\, g_1(x)) + (1-\tau) \log p_w(y \,|\, g_2(x)).$$

Compute the two entropy terms as

$$
\begin{aligned}
h_{yw} &= -\frac{1}{b_u}\sum_{i=1}^{b_u}\sum_{k=1}^{K}\frac{1}{K}\sum_{j=1}^{n}\sum_{y\in\mathcal{Y}} f(x^u_{ij}, y, w_k) f_{\log}(x^u_{ij}, y, w_k),\\
h_y &= -\frac{1}{b_u}\sum_{i=1}^{b_u}\sum_{y^n\in\mathcal{Y}^n}\left(\frac{1}{K}\sum_{k=1}^{K}\prod_{j=1}^{n} f(X^n_{ij}, y^n_j, w)\right)\log\left(\frac{1}{K}\sum_{k=1}^{K}\prod_{j=1}^{n} f(X^n_{ij}, y^n_j, w)\right).
\end{aligned}
$$

Compute the loss ℓ as

$$
\begin{aligned}
\ell_u &= \alpha h_y - h_{yw}\\
\ell_x &= -\frac{1}{bK}\sum_{i=1}^{b}\sum_{k=1}^{K}\log(p_{w_k}(y_i \,|\, x_i))\\
\ell &= \ell_x - \left(\frac{1}{1-\tau^2}\right)\ell_u.
\end{aligned}
$$

4.6.3. **Unsupervised transfer learning on MNIST**

For the following experiments on MNIST, the reference prior is of order $n = 2$ and has $L = 50$ particles. We run our methods for 1024 epochs.

We first compare deep reference priors with fine-tuning for transfer learning. The parameter β controls the degree to which the posterior (4.12) is influenced by the target data. If we have $\beta = 1$, then the posterior is maximally influenced by target data after being pretrained on the source data. We instantiate (4.12), by combining prior selection, pre-training on the source task into one objective,

$$
\max\nolimits_\pi \gamma I_\pi(w; Y^n, X^n) + (1-\beta)\mathbb{E}_{w\sim\pi}\log p(w; y^s \,|\, x^s), \tag{4.17}
$$

where γ and β are hyper-parameters. Solving (4.17) requires no knowledge from target data labels, therefore the setting here is pure unsupervised clustering for target task dataset. We compare this objective to fine-tuning which adapts a model trained on labeled source to the labeled target data. In this experiment, all samples from the source task (about 30,000 images across 5 classes) were used for both the reference prior and fine-tuning.

Method # Labeled target data (→)	0	50	100	250	500
Source (0–4) to Target (5–9)					
Fine-Tuning	-	71.1	78.8	86.6	93.0
Deep Reference Prior Unsupervised Transfer	87.4	-	-	-	-
Source (5–9) to Target(0–4)					
Fine-Tuning	-	90.2	92.4	94.7	96.2
Deep Reference Prior Unsupervised Transfer	95.2	-	-	-	-

Table 4.5: **Accuracy (%) of unsupervised reference-prior based transfer** (digits 0–4) to the target task (digits 5–9). We see that transfer using source and unlabeled target data using the reference prior performs as well as fine tuning with labeled source data and 250 labeled target data. Even if MNIST is a simple dataset, this is a remarkable demonstration of how effective the reference prior is at making use of both the labeled source data and unlabeled target data.

4.6.4. More ablation studies

Sec. 4.3.1 describes a few implementation tricks that we employ when computing $H(Y^n \,|\, X^n, w)$. The unlabeled samples consist of both weak and strong augmentations of the same image x which we denote by $g_1(x)$ and $g_2(x)$ and we define $p_{g_i} \equiv p_w(y|g_i(x), w)$. The objective can be upper-bounded using Jensen's inequality as follows

$$\begin{aligned} H(Y^n \,|\, X^n, w) &= -\mathbb{E}_{X^n} \int d(x|z) Y^n p_G(Y^n \,|\, X^n, w) \left[\log(p_G(Y^n \,|\, X^n, w))\right] \\ &= -\mathbb{E}_{X^n} \int d(x|z) Y^n p_G(Y^n \,|\, X^n, w) \left[\log(\tau p_{g_1} + (1-\tau) p_{g_2})\right] \\ &\leq -\mathbb{E}_{X^n} \int d(x|z) Y^n p_G(Y^n \,|\, X^n, w) \left[\tau \log p_{g_1} + (1-\tau) \log p_{g_2}\right] \end{aligned}$$

The first trick is to use the above bound from Jensen's inequality to compute $H(Y^n \,|\, X^n, w)$. The second trick we employ is to not update $p(y \,|\, g_1(x), w)$ with back-propagation gradients. Table 4.6 shows that both these tricks are needed to achieve good accuracy.

The third trick is to include x in the loss only if $\max p_w(y \,|\, g_1(x), w) > 0.95$ – an implementation detail also employed in Sohn et al. (2020). Table 4.6 shows that this has very little impact on accuracy.

Implementation trick	Accuracy (%)
Deep reference priors (All 3 tricks)	92.13
No stop gradient to p_{g_1}	10
No Jensen's inequality	86.55
No masking using probability threshold	92.35

Table 4.6: We perform an ablation study over the three implementation tricks considered in Sec. 4.3.1 and compute the accuracy after removing each one of the tricks. The accuracy (%) is computed for 250 labeled samples, with 4 particles and using order 2.

4.6.5. **Two-stage experiment for coin tossing**

In Sec. 4.4, we consider a situation when we obtain data in two stages, first Z^m, and then Z^n. We propose a prior π^* (4.10) such that the posterior of the second stage makes the maximal use of the new n samples. In this section, we visualize π^* in the parameter space using a two-stage coin tossing experiment. Consider the estimation of the bias of a coin $w \in [0, 1]$ using two-stage $m + n$ trials. There are m trials in first stage and n trails in second stage. If z denotes the number of heads in total, we have $p(z \mid w) = w^z(1 - w)^{m+n-z}(m + n)!/(z!(m + n - z)!)$. We numerically find π^* for different values of m and n using the BA algorithm (Fig. 4.6 and Fig. 4.7).

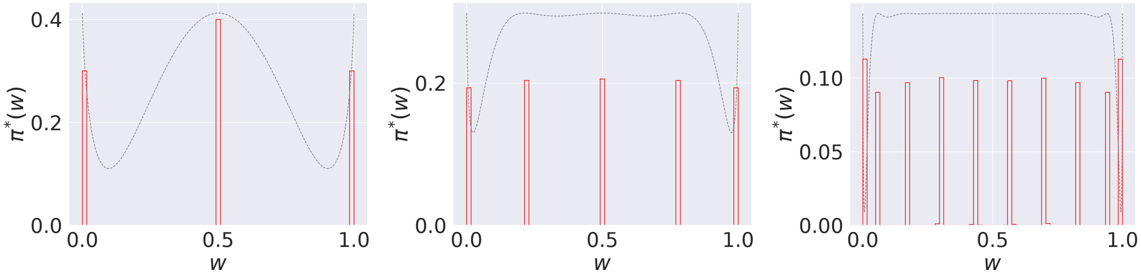


Figure 4.6: **Reference prior for the two stage coin-tossing model (see (4.10))** for $m = 1$ and $n = 1, 10, 40$ (from left to right) computed using the Blahut-Arimoto algorithm. Atoms are critical points of the gray line which is $\mathrm{KL}(p(Z^{m+n}), p(Z^{m+n} \mid w)) - \mathrm{KL}(p(Z^m), p(Z^m \mid w))$. The prior is again discrete for finite order $n < \infty$. We see how this reference prior behaves for different values of $\alpha = m/n$, e.g., for $\alpha \to 0$ this prior π^* is close to π_n^* in (4.3) but there are still some differences between them. This shows that the two-stage reference prior is not the same as the single-stage reference prior.

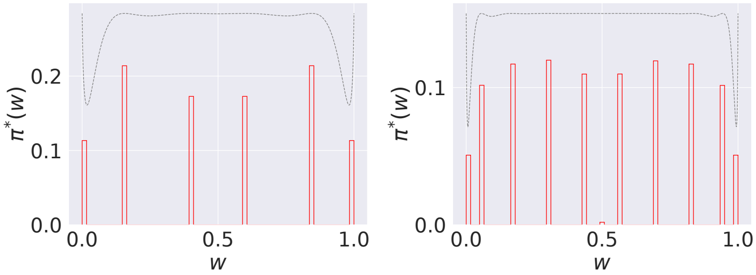


Figure 4.7: **Reference prior for the two stage coin-tossing model** (see (4.10)) for $n = 1$ and $m = 10, 30$ (from left to right) computed using the Blahut-Arimoto algorithm. Atoms are critical points of the gray line which is $\mathrm{KL}(p(Z^{m+n}), p(Z^{m+n} \,|\, w)) - \mathrm{KL}(p(Z^m), p(Z^m \,|\, w))$.

CHAPTER 5

A FOUNDATION PRIOR

This chapter introduces an information-geometric technique for analyzing the probabilistic models underlying deep neural networks. We present key information geometric concepts, including prediction space, divergence, infinitesimal distance, and visualization methods. Utilizing this new language allows us to interpret our previous results more simply and elegantly.

5.1. Information Geometry: Prediction Space, Metric, and Visualization

We represent deep neural networks as probabilistic models as they are trained to fit machine learning tasks. Consider a data set $D = \{(x_i, \hat{y}_i)\}_{i=1}^N$ of N independently and identically distributed samples, each sample consists of an input $x_i \in \mathcal{X}$ and the corresponding ground-truth label $\hat{y}_i \in \{1, 2, ..., C\}$, where C is the number of classes. We fix the measurement set $X = \{x_1, x_2, ..., x_N\}$ for the moment. Let $Y = \{y_1, y_2, ..., y_N\}$ denotes any set of possible outputs, each of which $y_i \in \{1, 2, ..., C\}$. For a network parameterized by $w \in \mathbb{R}^p$, we model the joint probability of the network predictions as

$$p_w(Y \,|\, X) := \prod_{i=1}^N p_w(y_i \,|\, x_i). \tag{5.1}$$

The probability distribution in 5.1 is $N \times C-$dimensional. Any network that makes predictions on the same measurement set X, irrespective of its architecture and the number of parameters, can be analyzed as a probabilistic model $p_w(\cdot \,|\, X)$ in this same $N \times C$-dimensional space; we will refer to this space as the *prediction space defined on the measurement set X*.

In the following chapters, we will conduct various experiments on deep learning models, including but not limited to model training, exploring the model's hypothesis class, adapting the model to new tasks through transfer learning, and traversing the possible hypothesis trained on diverse tasks. Throughout these processes, we will faithfully document the changes that occur in the models using the language of information geometry introduced in this chapter, without assumptions or embellishments, and try to derive general conclusions that can be widely applied to different deep learning systems. We hope that these conclusions serve as the foundation for a deeper understanding. In favor of the completeness of geometry, we define the metric, the divergence, the curves embedded in the prediction space, and the visualization techniques.

Measuring divergence in the prediction space We first mark a special point in the prediction space that we will refer to frequently. The true probabilistic model of the data, which corresponds to ground-truth labels, is denoted by $p_{\text{true}}(Y \,|\, X) := \prod_{i=1}^{N} \delta(y_i - \hat{y}_i)$, where $\hat{y}_i$ are ground-truth labels and δ is the Kronecker delta function. We will call this the *truth*. Given two probabilistic models p_u and p_v with weights u and v respectively, the Bhattacharyya divergence on the measurement set X between them is

$$\begin{aligned}
\mathrm{d_B}(p_u,\ p_v) &:= -\frac{1}{N} \log \sum_{\vec{y}} \prod_{i=1}^{N} \sqrt{p_u(y_i \,|\, x_i) \cdot p_v(y_i \,|\, x_i)} \\
&\overset{(*)}{=} -\frac{1}{N} \log \prod_{i=1}^{N} \sum_{c=1}^{C} \sqrt{p_u(c \,|\, x_i) \cdot p_v(c \,|\, x_i)} \\
&= -\frac{1}{N} \sum_{i=1}^{N} \log \sum_{c=1}^{C} \sqrt{p_u(c \,|\, x_i) \cdot p_v(c \,|\, x_i)}. \qquad (5.2)
\end{aligned}$$

Here $(*)$ follows because samples are independent. In other words, the Bhattacharyya divergence between two probabilistic models can be written as the average of the Bhattacharyya divergences of their predictive distributions $p_u(\cdot \,|\, x_i)$ and $p_v(\cdot \,|\, x_i)$ on each input x_i in the measurement set X.

Remark 29. One can also use other divergences to measure the discrepancy between P_u and P_v, such as the symmetrized Kullback-Leibler divergence,

$$\mathrm{d_{sKL}}(p_u,\ p_v) := \frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} (p_u(c \,|\, x_i) - p_v(c \,|\, x_i)) \log \frac{p_u(c \,|\, x_i)}{p_v(c \,|\, x_i)}, \qquad (5.3)$$

or the geodesic distance on the product space,

$$\mathrm{d_G}(p_u,\ p_v) := \frac{1}{N} \sum_{i=1}^{N} \arccos \sum_{c=1}^{C} \sqrt{p_u(c \,|\, x_i) \cdot p_v(c \,|\, x_i)}. \qquad (5.4)$$

Nevertheless, many other distances (e.g., Hellinger divergences) saturate quickly as the number of dimensions of the prediction space grows, obscuring the possible intrinsic low-dimensional structures. This is because two high-dimensional random vectors are orthogonal with high probability. When the number of samples N is large, distances such as the Bhattacharyya distance are better behaved due to their logarithms (Quinn et al., 2019b).

Measuring infinitesimal distance in the prediction space For two close networks, w and $w + \mathrm{d}w$, Bhattacharyya distance (5.2) induces a Riemannian structure. This allows us to write the infinitesimal distance $\mathrm{d}s$ in the prediction space,

$$\begin{aligned} \mathrm{d}s^2 &= 2\mathrm{KL}(p_w,\ p_{w+\mathrm{d}w}) \\ &= 4\mathrm{d}_{\mathrm{B}}(p_w,\ p_{w+\mathrm{d}w}) \\ &= \frac{1}{2}\mathrm{d}w^T \left(\frac{1}{N} \sum_{i,c} \nabla_w \log p_w(c \,|\, x_i) \nabla_w \log p_w(c \,|\, x_i)^T \right) \mathrm{d}w, \end{aligned} \tag{5.5}$$

notice that $\frac{1}{N}\sum_{i,c} \nabla_w \log p_w(c \,|\, x_i) \nabla_w \log p_w(c \,|\, x_i)^T$ is Fisher-Information Matrix (FIM) Amari (2016a). Weights w are a coordinate system for computing the distance. The FIM is the Hessian of the Bhattacharyya divergence; we may think of the FIM as quantifying the amount of information present in the model about the data it was trained on.

Measuring length of curves in the prediction space Given a continuously differentiable curve $\{w(t)\}_{t\in[0,1]}$ of network weights. We can compute the length of its corresponding curves in the prediction space by integrating the infinitesimal distance $|\mathrm{d}s|$ along it,

$$\int_0^1 \sqrt{2\mathrm{KL}(p_{w(t)},\ p_{w(t+\mathrm{d}t)})} = \int_0^1 \sqrt{4\mathrm{d}_{\mathrm{B}}(p_{w(t)},\ p_{w(t+\mathrm{d}t)})}. \tag{5.6}$$

The shortest length curve connecting two networks $u, v \in \mathbb{R}^p$ induces a metric known as the Fisher-Rao distance Radhakrishna Rao (1945),

$$\mathrm{d}_{\mathrm{FR}}(p_u,\ p_v) := \min_{w(t):w(0)=u,\ w(1)=v} \int_0^1 \sqrt{4\mathrm{d}_{\mathrm{B}}(p_{w(t)},\ p_{w(t+\mathrm{d}t)})}, \tag{5.7}$$

The shortest paths are geodesics, i.e., locally "straight lines".

Mapping a model trained on one task to another task using *imprinting*. This dissertation will consider different tasks $\{T^k\}_{k=1,\dots,}$ with the same input domain but possibly different numbers of classes C^k. Given a model $p^1_w(y \,|\, x)$ parametrized by weights w for task T^1, we want to evaluate its representation on another task, say, T^2. Let $w = (w_b, w_l)$ be the weights for the backbone and the linear classifier, respectively. The logits are ${w_l}^\top \varphi(x; w_b) \in \mathbb{R}^{C^1}$ corresponding to an input x and the penultimate layer features $\varphi(x; w_b)$. The output predictions $p^1_w(y \,|\, x_n)$ for $y = 1, \dots, C^1$ are computed using a Softmax applied to the logits. If we have learned w from one task T^1, we can re-initialize each row of the linear classifier weights $({w_l}')_y$ for $y = 1, \dots, C^2$ to maximize the cosine similarity with the average feature of samples from task T^2 with ground-truth class $\hat{y}$:

$$({w_l}')_y = h/\|h\|_2, \quad \text{where } h = \frac{\sum_{x \in T^2} \mathbb{1}_{\{\hat{y}(x)=y\}} \varphi(x; w_b)}{\sum_{x \in T^2} \mathbb{1}_{\{\hat{y}(x)=y\}}}, \tag{5.8}$$

$\mathbb{1}$ denotes the indicator function, x is randomly sampled from the task T^2, $\hat{y}(x)$ denotes the ground truth label of the input x. The new network $w = (w_b, {w_l}')$ can be used to predict T^2. Imprinting enables us to map a network trained on T^1 to another task T^2's prediction space.

Embedding a high-dimensional probabilistic model in lower dimensions. We use a visualization technique named *intensive principal component analysis (InPCA)* Quinn et al. (2019b) that embeds a probabilistic model into a lower-dimensional space. For r probability distributions $p_{w_1}, \dots, p_{w_r}$, consider a matrix $D \in \mathbb{R}^{r \times r}$ with entries $D_{ij} = \mathrm{d_B}(p_{w_i},\ p_{w_j})$ and

$$W = -LDL/2, \tag{5.9}$$

where $L_{ij} = \delta_{ij} - 1/r$ is the centering matrix. An eigendecomposition of $W = U \Sigma U^\top$ where the eigenvalues are sorted in descending order of their magnitudes $|\Sigma_{00}| \geq |\Sigma_{11}| \geq \dots$ allows us to compute the embedding of these r probability distributions into an r-dimensional space as $\mathbb{R}^{r \times r} \ni X = U\sqrt{\Sigma}$. Unlike standard PCA where eigenvalues are non-negative, eigenvalues of InPCA can be both positive and negative, i.e., the lower-dimensional space is a Minkowski space Quinn et al.

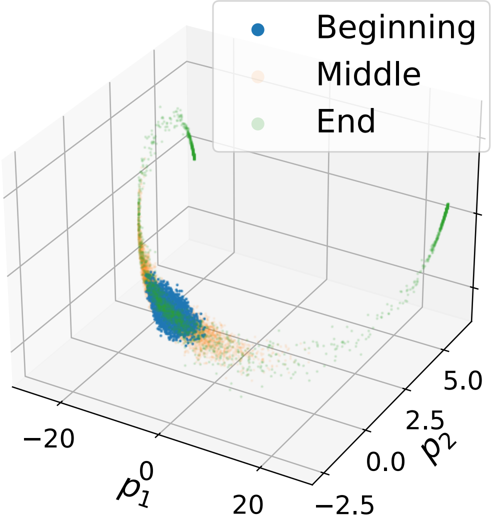


Figure 5.1: **Reference prior (green) for binary classification on MNIST**. A three-dimensional embedding of the probability distributions of $L = 3000$ atoms in the reference prior after 50,000 iterations of the BA algorithm (green) for a binary classification problem on MNIST (digits 3 vs. 5). Particles were initialized Gaussian (blue) randomly, and they are nearby with each other in this embedding because, at initialization, the logits of each particle are uniformly distributed. Orange shows particle locations after 5,000 iterations. As the reference prior objective in (4.3) is optimized, the particles increasingly make more distinguish predictions (orange), and towards the end (green), these particles spread apart boundaries of the hypothesis class.

(2019b). This allows the InPCA embedding to be an isometry, i.e., pairwise distances are preserved:

$$\textstyle\sum_{k=1}^{r}(X_{ki} - X_{kj})^2 = \mathrm{d_B}(p_{w_i},\ p_{w_j}) \geq 0 \tag{5.10}$$

for column embeddings $X_{:i},\ X_{:j}$ of two probabilistic models $p_{w_i},\ p_{w_j}$.

5.1.1. Visualizing the reference priors in the prediction space

We can adopt an information geometric perspective introduced in this section to characterize the reference priors in the prediction space. As we discussed in Chapter 4, reference priors reconstruct a diverse set of tasks at the boundaries of the hypothesis class based on the input data.

Given a $C-$ way classification task with input domain $\mathcal{X}$ and a family of neural networks parameterized by $w \in \mathbb{R}^p$, the hypothesis class $\mathcal{H}$ is a collection of candidate solutions that can be used to solve the given task. Each candidate solution in this hypothesis class is a prediction function $p_w(y \,|\, x)$

parameterized by w. Typically, the input domain $\mathcal{X}$ is a continuous space, and p_w is a function belonging to an infinite-dimensional functional space. This makes it challenging to work with. As in (5.1), we will map the hypothesis p_w to the probabilistic model $p_w(\cdot \mid X)$ in the prediction space defined on a finite measurement set $X = \{x_1, x_2, ...\} \subset \mathcal{X}$.

For each particles in Rem. 23, we compute its corresponding probabilistic model (5.1) ($p_w(\cdot \mid X^N)$) in the prediction space defined on a measurement set $X^N = \{x_1, ..., x_N\}$. We compute a principal component analysis (InPCA) of such probabilistic models $\{p_{w^1}(\cdot \mid X^N), ..., p_{w^L}(\cdot \mid X^N)\}$ using a method developed in Quinn et al. (2019a) and visualize the in Fig. 5.1.

5.1.2. **Measuring task distances in the prediction space**

In Chapter 3, we have focused on the process of transferring a pre-trained model to adapt to a new task while adopting an information geometric perspective to observe the model transfer process. In this section, we leverage the power of information geometric tools developed in Sec. 5.1 and re-formalize the information geometric distance on the space of tasks defined in Chapter 3.

We represent the infinite-dimensional object $p_w(y|x)$ by mapping it to a probabilistic model (5.1) in the prediction space. We next combine the development of measuring the length of curves in the prediction space (5.6) and optimal transportation for discrete measures (3.4). We transport the margin on the data and modify the model weights simultaneously. We call this method the *coupled transfer process* and the corresponding task distance as the *coupled transfer distance.*

Definition 30 (Uncoupled transfer distance). We first discuss a simple transport mechanism instead of OT and discuss how to compute a transfer distance. For $\tau \in [0, 1]$, consider the mixture distribution

$$\hat{p}_\tau(x, y) = (1 - \tau)\hat{p}_s(x, y) + \tau\hat{p}_t(x, y). \tag{5.11}$$

Samples from $\hat{p}_\tau$ can be drawn by sampling an input-output pair from $\hat{p}_s$ with probability $1-\tau$ and sampling it from $\hat{p}_t$ otherwise. At each time instant τ, the uncoupled transfer process updates the weights of the classifier using SGD to fit samples from $\hat{p}_\tau$

$$w(\tau + \mathrm{d}\tau) = w(\tau) - \hat{\nabla}\ell_\tau(w(\tau))\ \mathrm{d}\tau;\ w(0) = w_s. \tag{5.12}$$

Weights $w(\tau)$ are thus fitted to each task p_τ as τ goes from 0 to 1. In particular for $\tau = 1$, weights $w(1)$ are fitted to $\hat{p}_t$. As $\mathrm{d}\tau \to 0$, we obtain a continuous curve $\{w(\tau) : t \in [0, 1]\}$. Computing the length of this weight trajectory gives a transfer distance analogy to (5.6),

$$\int_0^1 \sqrt{\mathbb{E}_{x\sim\hat{p}_\tau} 2\mathrm{KL}\left[p_{w(\tau)}(\cdot\,|\,x), p_{w(\tau+\mathrm{d}\tau)}(\cdot\,|\,x)\right]}. \tag{5.13}$$

Definition 31 (Coupled transfer distance). Given two learning tasks D_s and D_t and a w-parametrized classifier trained on D_s with weights w_s, the coupled transfer distance between the tasks is

$$\min_{\Gamma, w(\cdot)} \mathbb{E}_{x\sim\hat{p}_\tau} \int_0^1 \sqrt{2\mathrm{KL}\left[p_{w(\tau)}(\cdot\,|\,x), p_{w(\tau+\mathrm{d}\tau)}(\cdot\,|\,x)\right]} \tag{5.14}$$

where and couplings $\Gamma \in \Pi(\hat{p}_s(x), \hat{p}_t(x))$ and $w(\cdot)$ is a continuous curve which is the limit of

$$w(\tau + \mathrm{d}\tau) = w(\tau) - \hat{\nabla}\ell_\tau(w(\tau))\ \mathrm{d}\tau;\ w(0) = w_s.$$

as $\mathrm{d}\tau \to 0$. The interpolated distribution $\hat{p}_\tau(x, y)$ at time instant $\tau \in [0, 1]$ for a coupling Γ is given by (3.4) and the loss ℓ_τ is the cross-entropy loss of fitting data from this interpolated distribution.

In comparison of (5.13), we move the expectation $\mathbb{E}_{x\sim\hat{p}_\tau}$ outside the square root for simplifying the computation.

5.2. Building a Foundation Prior: Mapping and Selecting Representative Experts in the Prediction Space

Currently, many researchers are pursuing the development of foundation models. However, statistical learning theory insights (e.g., potential *task competition*) suggest that building a single model for all tasks may not be ideal (Ramesh and Chaudhari, 2022a; Baxter, 2000; Hanneke and Kpotufe, 2022). **Instead, it is more appropriate to consider a mixture of experts selected based on priors instead of relying solely on an overconfident point estimator.**

In this section, we propose a mechanism to select a mixture of representative experts trained on typical learnable tasks, and the combined recorded expert models form a powerful prior known as the *foundation prior*. We also design an algorithm to utilize the foundation prior efficiently, and our experimental results demonstrate the algorithm's effectiveness. It is important to note that while foundation models may not be suitable for all tasks, the foundation prior, formally a mixture of experts, is expected to perform better.

We are interested in the supervised learning problem in this chapter. Consider a labeled dataset $\{(x_i, \hat{y}_i)\}_{i=1}^N$ of N samples, each of which consists of an input x_i sampled from the given task and its corresponding ground-truth label $\hat{y}_i \in \{1, 2, ..., C\}$, where C is the number of classes. Let $w \in \mathbb{R}^p$ denote the parameters of the network. Let $\{x_{N+1}, x_{N+2} ..., x_{N+M}\}$ be an unlabelled data we are interested in making predictions on, each of which is sampled from the same given task. Given a prior on the parameters $\pi(w)$, Bayes law gives the posterior

$$p\left(w \mid \{(x_i, \hat{y}_i)\}_{i=1}^N\right) \propto \pi(w) \prod_{i=1}^N p_w(\hat{y}_i \,|\, x_i).$$

Here we assume data are independently and identically sampled from the task. The choice of a prior is based on information we believe in prior to observing the ground truth labels. The simplest and oldest rule for determining a prior is the *principle of indifference*, which assigns equal probabilities to all hypotheses. If the input dataset $\{x_i\}_{i=1}^{N+M}$ and the network architecture represent all our knowledge about the task prior to observing the labels, Jeffreys prior $\pi_J(w)$ assigns the equal probability to a set of hypothesis $p_w(y \,|\, x)$ irrespective of our choice of network parameterization and satisfies the principle of indifference,

$$\pi_J(w) \propto \sqrt{\det F(w)}, \tag{5.15}$$

where $F(w)$ is *Fisher information matrix (FIM)*,

$$F(w) = \frac{1}{N+M} \sum_{i=1}^{N+M} \sum_{y} p_w(y \,|\, x) \nabla_w \log p_w(y \,|\, x) \nabla_w \log p_w(y \,|\, x)^T. \tag{5.16}$$

However, recent research (Mao et al., 2023) suggests that even if the size of the hypothesis class might be very complex for deep networks (Dziugaite and Roy, 2017; Bartlett et al., 2017), the subset of the hypothesis space explored by typical deep learning algorithms might be much smaller and low-dimensional. The emergence of the low-dimensional hypothesis class might arise from (a) the structure of typical datasets (Goldt et al., 2020; d'Ascoli et al., 2021; Refinetti et al., 2021; Yang et al., 2022), e.g., spectral properties, and (b) the models initialization (Dziugaite and Roy, 2017; Bartlett et al., 2017).

We consider the emergence of low dimensionality as a component of our universal prior knowledge, which suggests that considering the entire hypothesis class may not be necessary. Instead, we prefer building a prior that only assigns equal probability to all possible hypotheses explored by the typical deep learning tasks and algorithms. We refer to this as *a foundation prior*. This is part of our primary motivations for writing this chapter.

In this chapter, we represent all possible infinite-dimensional objects $p_w(y \,|\, x)$, as explored by typical learnable tasks, by mapping them onto the probabilistic models in the prediction space. These probabilistic models form a low-dimensional manifold within the prediction space, thereby enabling the construction of an effective foundational prior. We show an empirical study of the foundation prior to the CIFAR-10 and CIFAR-100 datasets.

5.2.1. Possible expert models explored by typical learnable tasks

Let $\mathcal{T} = \left\{T^1,\, T^2,\, ...,\, T^K\right\}$ denote a collection of the typical task dataset, each of which T^k represents a dataset of a specific task. We expect to maximize the number and diversity of the tasks within $\mathcal{T}$ as much as possible. We assume all tasks in $\mathcal{T}$ have the same input domain but possibly a different number of output classes $C^1,\ C^2,\ ....$ We collect N_k training samples for the task T^k.

We aim to glean a global picture of all possible models p_w explored by typical learnable tasks in $\mathcal{T}$. Even though the models may be trained on different tasks, we implement imprinting (5.8) to map them onto the probabilistic models in the same prediction space with dimension $N \times C$, where $N = \sum_k N_k$ and $C = \sum_k C_k$. However, finding all possibilities that meet our needs is still a challenging problem. Miscellaneous factors such as initialization, training progresses, and the different combinations of the selected tasks in $\mathcal{T}$ lead to diverse representations. We design the following random search algorithm Definition 32 to overcome these challenges and traverse the possible hypothesis.

Definition 32 (Random searching the possibility.). We randomly sample a task from $\mathcal{T}$ at the beginning stage. The model is then trained to fit the selected task. In each subsequent iteration of the optimization algorithm, we have a small chance $0 < p < 1$ to switch the task. Whenever a task switching occurs, the loss function is adjusted as follows: the current loss function is computed as a weighted combination of the new task's loss function before the switch, with equal weights of 0.5 assigned to each term. This adjustment ensures a balanced consideration of both the new and previously encountered tasks during the optimization process. The new task is also randomly sampled from $\mathcal{T}$. This allows the training trajectory to adapt to random tasks and traverse the possible hypothesis continually. We run S training iterations and record 200 checkpoints for later analysis. We repeat this procedure for multiple times. Using imprinting (5.8), we map all saved checkpoints onto probabilistic models in $N \times C$ dimension prediction space.

In this section, we execute Definition 32 and obtain a general view of the possible models trained on typical learnable tasks within CIFAR100 (Krizhevsky, 2009). The CIFAR100 dataset has 100 classes containing 500 training images each. The 100 classes in the CIFAR-100 are grouped into 20 super-classes, i.e., the super-class named flowers consists of five subordinate classes: orchids, poppies, roses, sunflowers, and tulips. Each super-class in CIFAR100 represents a $5-$way classification task. Therefore, CIFAR10 is a collection of 20 individual vision tasks.

We randomly pick a super-class from CIFAR100 at the beginning stage. The model is then trained to fit the selected task. We execute 1000 training epochs and record 200 checkpoints for later

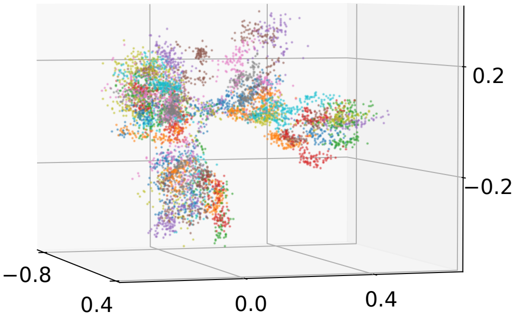


Figure 5.2: This figure shows the possible expert models trained on typical learnable tasks within CIFAR100, visualized by InPCA. We executed Definition 32 multiple times and recorded 12800 checkpoints. Using imprinting (5.8), we mapped all saved checkpoints onto probabilistic models within a $50,000 \times 100$ dimensional prediction space. Then, we computed the InPCA embedding for all probabilistic models and selected the top-3 leading eigenvalues to derive the corresponding 3 eigen-directions for visualization. The numerical values on each axis represent the projection of the InPCA embedding along the respective eigen-directions. Each dot in this figure represents a probabilistic model. The multiple executions of Definition 32 resulted in diverse explorations in the prediction space, primarily due to the randomness. However, the probabilistic models trained on various combinations of tasks consistently occupy a low-dimensional subspace within the prediction space.

analysis. In each subsequent epoch of the optimization algorithm, we have a probability 0.02 to switch the task. Whenever a task switching occurs, the current loss function is computed as a weighted combination of the loss function on the new task and the loss function before the switch, with equal weights of 0.5 assigned to each term as discussed in Definition 32. We repeat this procedure for 64 times. Using imprinting, we map all 64×200 saved checkpoints onto probabilistic models within $50,000 \times 100$ dimension prediction space. Then we compute the InPCA embedding for all probabilistic models. We pick the 3 eigen-directions corresponding to top-3 leading eigenvalues and visualize the InPCA embedding in Fig. 5.2.

5.2.2. Max-min divergence sampling selects representative experts trained on typical learnable tasks

The foundation prior assigns equal probability to all possible models (i.e., dots in Fig. 5.2) explored by the typical learnable tasks (e.g., tasks within CIFAR100). There are $64 \times 200 = 12,800$ probabilistic models visualized in Fig. 5.2. Therefore building a foundation prior might be quite expensive. To make the foundation prior practical, we select a group of representative probabilistic models using a *Max-Min Divergence Sampling* algorithm.

Definition 33 (Max-Min Divergence Sampling). Max-Min Divergence Sampling is a sub-sampling technique commonly used in data selection. In Definition 32, we mapped all saved models onto probabilistic models in $N \times C$ dimension prediction space using imprinting. The Max-Min Divergence Sampling algorithm begins by computing the pairwise Bhattacharyya divergence (5.2) D_{ij} between the saved checkpoints parameterized by w_i and w_j, respectively,

$$D_{ij} = -\frac{1}{N} \sum_{x} \log \sum_{y} \sqrt{p_{w_i}(y \,|\, x) \cdot p_{w_j}(y \,|\, x)},$$

for $1 \leq i,\ j \leq 12,800$. Next, the algorithm iteratively selects checkpoints that have the maximum minimum Bhattacharyya divergence to their nearest neighbors. The algorithm identifies the checkpoint with the largest minimum Bhattacharyya divergence to its nearest neighbor among the remaining unselected points in each iteration. This checkpoint is then added to the sub-sample set. We give the pseudo-code for the algorithm in Algorithm 1.

Algorithm 1 Max-Min Divergence Sampling

```
Require: Pairwise Bhattacharyya divergence $D_{ij}$, $1 \leq i,\ j \leq 12,800$; Desired sub-sample size $K$
Ensure: Sub-sample $\mathcal{S}$ of size $K$
  $\mathcal{S} \leftarrow \{\ \}$                                  ▷ Initialize an empty sub-sample $\mathcal{S}$
  $i_1 \sim \text{Uniform}(\{1, 2, ..., 12800\})$                  ▷ Randomly select index $i_1$
  $\mathcal{S} \leftarrow \mathcal{S} \cup \{i_1\}$                ▷ Add $i_1$ to $S$
  for $k = 2$ to $K$ do
      for $i = 1$ to 12800 do                                      ▷ Iterate over the saved 12800 checkpoints
          $d_i^{\min} \leftarrow \min\{D_{ij} \,|\, j \in \mathcal{S}\}$   ▷ Calculate the minimum divergence $d_i^{\min}$
      end for
      $i_k \leftarrow \arg\max_i \{d_i^{\min}\}$                   ▷ Select $i_k$ with the maximum minimum divergence
      $\mathcal{S} \leftarrow \mathcal{S} \cup \{i_k\}$            ▷ Add $i_k$ to $\mathcal{S}$
  end for
      return Sub-sample $\mathcal{S}$
```

We select 100 representative checkpoints from Fig. 5.2 using Algorithm 1, as shown in Fig. 5.3

By selecting checkpoints with the maximum minimum divergence, Algorithm 1 ensures that the chosen sub-samples are well spread out across the probabilistic models manifold, see Fig. 5.3. This

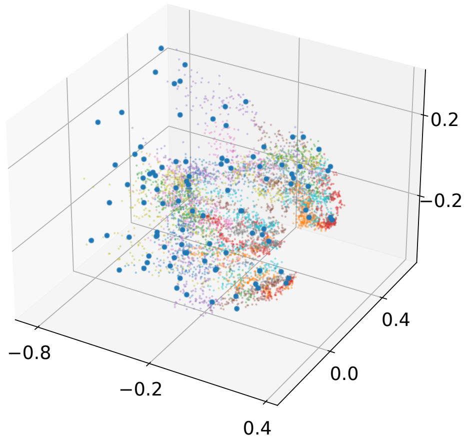


Figure 5.3: **A Prototype of Foundation Prior:** Max-Min Divergence Sampling Algorithm 1 selects 100 representative experts models from the saved checkpoints in Fig. 5.2. The small dots represent the originally saved probabilistic models visualized in Fig. 5.2, while the giant blue dots represent our sub-sampled experts. Algorithm 1 ensures that the chosen sub-samples are well spread out across the probabilistic models manifold. This helps to capture the diversity and preserve the overall structure.

helps to capture the diversity and preserve the overall structure.

5.2.3. **Foundation prior selects solutions from the hypothesis class**

Let $\mathcal{S} = \{i_1, i_2, .., i_K\}$ denote a collection of the saved checkpoint indices selected by Algorithm 1. Each chosen checkpoint is parameterized by w_{i_k}, for $1 \leq k \leq K$. We aim to address a task together with a labeled dataset $\{(x_i, \hat{y}_i)\}_{i=1}^{N}$ of N samples and a validation dataset $\{x_{N+1}, x_{N+2}..., x_{N+M}\}$ that we are interested in making predictions on. Let $q(y \,|\, x)$ denote a possible hypothesis. To make the foundation prior assign equal probabilities to the possible solutions spanned by the checkpoints in Σ, we define the foundation prior π_F on the hypothesis class as,

$$\pi_F(q) \propto \exp\left\{-\frac{1}{|\mathcal{S}|}\sum_{k\in\mathcal{S}}\frac{1}{N+M}\sum_{i=1}^{N+M}\mathrm{d_B}\left[q(\cdot\,|\,x_i),\; p_{w_k}(\cdot\,|\,x_i)\right]\right\}, \tag{5.17}$$

where $\mathrm{d_B}\left[q(\cdot \,|\, x_i),\ p_{w_k}(\cdot \,|\, x)\right] = -\log \sum_y \sqrt{q(y \,|\, x) \cdot p_{w_k}(y \,|\, x)}$ denotes Bhattacharyya divergence. π_F is a prior defined on the functional space. The functional Bayesian rule gives the posterior on q,

$$p\left(q \,\middle|\, \{(x_i, \hat{y}_i)\}_{i=1}^{N},\ \{x_j\}_{j=N+1}^{N+M}\right) = \prod_{i=1}^{N} q\left(\hat{y}_i \,|\, x_i\right) \pi_F(q). \tag{5.18}$$

5.3. Experiments

We sub-sample 100 checkpoints from Fig. 5.2 using Algorithm 1. These checkpoints form the building blocks of the foundation prior; see Sec. 5.2.3. We evaluate the foundation prior π_F on CIFAR-10 (Krizhevsky, 2009). CIFAR-10 has $50,000$ labeled data in the training set. We only use 100–10,000 labeled samples, i.e., 10–1,000 samples/class, and use the samples in the test set as unlabeled samples that we are interested in making predictions on. All experiments use the WRN 16-4 architecture (Zagoruyko and Komodakis, 2016). All experiments use weak data augmentations such as random horizontal flips and random crops in the training stage.

The posterior (5.18) suggests that we should first compute the prior and then weight solutions by the likelihood of the labeled data. In practice, we combine these two steps into a single objective

$$\min_w -\frac{1}{N}\sum_{i=1}^{N} \log p_w(\hat{y}_i \,|\, x_i) + \gamma \frac{1}{|\mathcal{S}|} \sum_{k \in \mathcal{S}} \frac{1}{N+M} \sum_{i=1}^{N+M} \mathrm{d_B}\left[p_w(\cdot \,|\, x_i),\ p_{w_k}(\cdot \,|\, x_i)\right], \tag{5.19}$$

Where γ is a hyperparameter.

Baselines We compare the foundation prior to classical supervised learning.

5.4. Discussion

Statistical learning theory (Ramesh and Chaudhari, 2022a; Baxter, 2000; Hanneke and Kpotufe, 2022) recognizes the absence of a universal solution in multi-task learning, where building a single model for all tasks often leads to compromised performance on specific tasks as task diversity increases. If a diverse set of tasks with average task distances beyond a certain threshold exists, the free lunch theorem for multi-task learning will surely come into play. This could be the point at which foundation models fail. The foundation priors address this issue by selecting representative

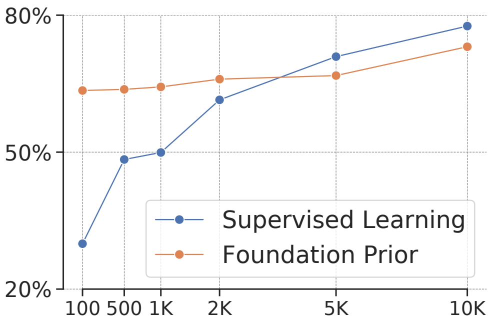


Figure 5.4: We compare Bayesian inference with a foundation prior (5.19) and supervised learning on the CIFAR-10 dataset. Our findings reveal that the foundation prior consistently outperforms supervised learning in the small data regime, specifically with 100–2,000 labeled samples. This experimental evidence demonstrates the efficacy of the foundation prior approach for learning from limited data.

experts trained on typical learnable tasks. It is important to note that while foundation models may not be suitable for all tasks, the foundation prior, formally a mixture of experts, is expected to yield superior performance.

CHAPTER 6

CONCLUSION

Statistical learning theory (Ramesh and Chaudhari, 2022a; Baxter, 2000; Hanneke and Kpotufe, 2022) recognizes the absence of a universal solution in multi-task learning, where building a single model for all tasks often leads to compromised performance on specific tasks as task diversity increases. However, recent advancements in models such as GPT and CLIP have demonstrated the practical feasibility of fine-tuning and adapting them to a wide range of diverse tasks. This apparent contradiction serves as the partial motivation for our research. Our hypothesis posits that typical tasks addressed by current researchers are sufficiently similar to each other, to the extent that the free lunch theorem for multi-task learning might fail to fully explain. Recent research has provided evidence for this hypothesis by revealing the emergence of low dimensionality in the space of learnable tasks (Mao et al., 2023; Ramesh et al., 2022). However, we need to precisely and mathematically formalize and substantiate this hypothesis.

This dissertation identifies reconstruction as the canonical task that pre-training procedures should consider beneficial to multiple downstream tasks, harnesses the power of reconstruction in pre-training with unlabeled data, and arises an information geometric correct pairwise task distances. However, the global landscape of the space of typical learnable tasks remains unexplored in this dissertation. Is the diameter of typical learnable task space smaller than expected or only slightly larger than the pairwise task distances?

On the other hand, if a diverse set of tasks with average task distances beyond a certain threshold exists, the free lunch theorem for multi-task learning will surely come into play. This could be the point at which foundation models fail. The foundation priors address this issue by selecting representative experts trained on typical learnable tasks. It is important to note that while foundation models may not be suitable for all tasks, the foundation prior, formally a mixture of experts, is expected to yield superior performance.

BIBLIOGRAPHY

Ape – home page. http://ape-package.ird.fr/.

Michael C. Abbott and Benjamin B. Machta. A Scaling Law From Discrete to Continuous Solutions of Channel Capacity Problems in the Low-Noise Limit. *Journal of Statistical Physics*, 176(1):214–227, July 2019. ISSN 1572-9613. doi: 10.1007/s10955-019-02296-2.

Alessandro Achille and Stefano Soatto. On the emergence of invariance and disentangling in deep representations. *arXiv:1706.01350*, 2017.

Alessandro Achille, Glen Bigan Mbeng, and Stefano Soatto. The dynamic distance between learning tasks.

Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless Fowlkes, Stefano Soatto, and Pietro Perona. Task2vec: Task embedding for meta-learning. *arXiv preprint arXiv:1902.03545*, 2019a.

Alessandro Achille, Glen Mbeng, and Stefano Soatto. Dynamics and Reachability of Learning Tasks. *arXiv:1810.02440 [cs, stat]*, May 2019b.

Alessandro Achille, Giovanni Paolini, Glen Mbeng, and Stefano Soatto. The Information Complexity of Learning Tasks, their Structure and their Distance. *arXiv:1904.03292 [cs, math, stat]*, April 2019c.

Alexander A Alemi. Variational predictive information bottleneck. In *Symposium on Advances in Approximate Bayesian Inference*, pages 1–6. PMLR, 2020.

Alexander A Alemi and Ian Fischer. Therml: Thermodynamics of machine learning. *arXiv preprint arXiv:1807.04162*, 2018.

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

Alexander A Alemi, Ben Poole, Ian Fischer, Joshua V Dillon, Rif A Saurous, and Kevin Murphy. Fixing a broken elbo. *arXiv preprint arXiv:1711.00464*, 2017.

David Alvarez-Melis and Nicolò Fusi. Geometric dataset distances via optimal transport. *arXiv preprint arXiv:2002.02923*, 2020a.

David Alvarez-Melis and Nicolò Fusi. Gradient flows in dataset space. *arXiv preprint arXiv:2010.12760*, 2020b.

David Alvarez-Melis and Tommi Jaakkola. Gromov-Wasserstein Alignment of Word Embedding

Spaces. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1881–1890, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1214.

Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016a.

Shun-ichi Amari. *Information Geometry and Its Applications*, volume 194 of *Applied Mathematical Sciences*. Springer Japan, Tokyo, 2016b. ISBN 978-4-431-55977-1 978-4-431-55978-8. doi: 10.1007/978-4-431-55978-8.

Suguru Arimoto. An algorithm for computing the capacity of arbitrary discrete memoryless channels. *IEEE Transactions on Information Theory*, 18(1):14–20, 1972.

Vijay Balasubramanian. Statistical inference, occam's razor, and statistical mechanics on the space of probability distributions. *Neural computation*, 9(2):349–368, 1997.

Peter L. Bartlett and Shahar Mendelson. Rademacher and Gaussian Complexities: Risk Bounds and Structural Results. In G. Goos, J. Hartmanis, J. van Leeuwen, David Helmbold, and Bob Williamson, editors, *Computational Learning Theory*, volume 2111, pages 224–240. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. ISBN 978-3-540-42343-0 978-3-540-44581-4. doi: 10.1007/3-540-44581-1_15.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*, pages 6240–6249, 2017.

Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

Martin Bauer, Martins Bruveris, and Peter W. Michor. Uniqueness of the Fisher–Rao metric on the space of smooth densities. *Bulletin of the London Mathematical Society*, 48(3):499–506, June 2016. ISSN 0024-6093. doi: 10.1112/blms/bdw020.

Heinz H. Bauschke and Patrick L. Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. CMS Books in Mathematics. Springer International Publishing, second edition, 2017. ISBN 978-3-319-48310-8. doi: 10.1007/978-3-319-48311-5.

Jonathan Baxter. A model of inductive bias learning. *Journal of artificial intelligence research*, 12:149–198, 2000.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.

James O Berger, Jos M Bernardo, and Manuel Mendoza. *On Priors That Maximize Expected Information*. Purdue University. Department of Statistics, 1988.

James O Berger, José M Bernardo, and Dongchu Sun. The formal definition of reference priors. *The Annals of Statistics*, 37(2):905–938, 2009.

Jose M Bernardo. Reference posterior distributions for Bayesian inference. *Journal of the Royal Statistical Society: Series B (Methodological)*, 41(2):113–128, 1979.

David Berthelot, Nicholas Carlini, Ekin D Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. *arXiv preprint arXiv:1911.09785*, 2019a.

David Berthelot, Nicholas Carlini, Ian Goodfellow, Nicolas Papernot, Avital Oliver, and Colin A Raffel. MixMatch: A holistic approach to semi-supervised learning. *Advances in Neural Information Processing Systems*, 32, 2019b.

William Bialek, Ilya Nemenman, and Naftali Tishby. Predictability, complexity, and learning. *Neural computation*, 13(11):2409–2463, 2001.

Richard Blahut. Computation of channel capacity and rate-distortion functions. *IEEE transactions on Information Theory*, 18(4):460–473, 1972.

Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Summer School on Machine Learning*, pages 169–207. Springer, 2003.

Rob Brekelmans, Daniel Moyer, Aram Galstyan, and Greg Ver Steeg. Exact rate-distortion in autoencoders via echo noise. In *Advances in Neural Information Processing Systems*, pages 3884–3895, 2019.

Guillaume Carlier, Alfred Galichon, and Filippo Santambrogio. From Knothe's transport to Brenier's map and a continuation method for optimal transport. *SIAM Journal on Mathematical Analysis*, 41(6):2554–2576, 2010.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Pratik Chaudhari and Stefano Soatto. Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks. *arXiv preprint arXiv:1710.11029*, 2017.

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. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12): 124018, 2019.

Bertrand S Clarke and Andrew R Barron. Jeffreys' prior is asymptotically least favorable under entropy risk. *Journal of Statistical planning and Inference*, 41(1):37–60, 1994.

Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020.

Yin Cui, Yang Song, Chen Sun, Andrew Howard, and Serge Belongie. Large scale fine-grained categorization and domain-specific transfer learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4109–4118, 2018.

Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. In *Advances in neural information processing systems*, pages 2292–2300, 2013.

Stéphane d'Ascoli, Marylou Gabrié, Levent Sagun, and Giulio Biroli. On the interplay between data structure and loss function in classification problems. *Advances in Neural Information Processing Systems*, 34:8506–8517, 2021.

Christos Davatzikos. Machine learning in neuroimaging: Progress and challenges. *NeuroImage*, 197: 652, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina N. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. 2018.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL HLT*, 2019.

Guneet S Dhillon, Pratik Chaudhari, Avinash Ravichandran, and Stefano Soatto. A baseline for few-shot image classification. In *Proc. of International Conference of Learning and Representations (ICLR)*, 2020.

Carl Doersch and Andrew Zisserman. Multi-task self-supervised visual learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2051–2060, 2017.

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.

Kilian Fatras, Younes Zine, Rémi Flamary, Rémi Gribonval, and Nicolas Courty. Learning with minibatch wasserstein : asymptotic and gradient properties, 2020.

David J Field. What is the goal of sensory coding? *Neural computation*, 6(4):559–601, 1994.

Yansong Gao and Pratik Chaudhari. A free-energy principle for representation learning. In *Proc. of International Conference of Machine Learning (ICML)*, 2020a.

Yansong Gao and Pratik Chaudhari. A free-energy principle for representation learning, 2020b.

Yansong Gao and Pratik Chaudhari. An information-geometric distance on the space of tasks, 2021.

Yansong Gao, Rahul Ramesh, and Pratik Chaudhari. Deep reference priors: What is the best way to pretrain a model?, 2022.

Bernhard C Geiger. On information plane analyses of neural network classifiers–a review. *IEEE Transactions on Neural Networks and Learning Systems*, 2021.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

Ziv Goldfeld, Ewout van den Berg, Kristjan Greenewald, Igor Melnyk, Nam Nguyen, Brian Kingsbury, and Yury Polyanskiy. Estimating information flow in deep neural networks. *arXiv preprint arXiv:1810.05728*, 2018.

Sebastian Goldt, Marc Mézard, Florent Krzakala, and Lenka Zdeborová. Modeling the influence of data structure on learning in neural networks: The hidden manifold model. *Physical Review X*, 10(4):041044, 2020.

Sarah C Goslee, Dean L Urban, et al. The ecodist package for dissimilarity-based analysis of ecological data. *Journal of Statistical Software*, 22(7):1–19, 2007.

Yves Grandvalet, Yoshua Bengio, et al. Semi-supervised learning by entropy minimization. *CAP*, 367:281–296, 2005.

Steve Hanneke and Samory Kpotufe. A no-free-lunch theorem for multitask learning. *The Annals of Statistics*, 50(6):3119–3143, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *arXiv:1603.05027*, 2016a.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *arXiv:1603.05027*, 2016b.

I Higgins, L Matthey, A Pal, C Burgess, X Glorot, M Botvinick, S Mohamed, and Lerchner A. beta-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework . In *ICLR*, 2017.

Kyle Hsu, Sergey Levine, and Chelsea Finn. Unsupervised learning via meta-learning. *arXiv preprint arXiv:1810.02334*, 2018.

Junlin Hu, Jiwen Lu, and Yap-Peng Tan. Deep transfer metric learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 325–333, 2015.

S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv:1502.03167*, 2015.

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

Michael I Jordan, Zoubin Ghahramani, Tommi S Jaakkola, and Lawrence K Saul. An introduction to variational methods for graphical models. In *Learning in graphical models*, pages 105–161. Springer, 1998.

Armand Joulin, Laurens van der Maaten, Allan Jabri, and Nicolas Vasilache. Learning Visual Features from Large Weakly Supervised Data. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, Lecture Notes in Computer Science, pages 67–84, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46478-7. doi: 10.1007/978-3-319-46478-7_5.

Byoungjip Kim, Jinho Choo, Yeong-Dae Kwon, Seongho Joe, Seungjai Min, and Youngjune Gwon. Selfmatch: Combining contrastive self-supervision and consistency for semi-supervised learning. *arXiv preprint arXiv:2101.06480*, 2021.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. *arXiv:1312.6114*, 2013.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. *arXiv:1312.6114 [cs, stat]*, May 2014.

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*, 114(13):3521–3526, 2017.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Large scale learning of general visual representations for transfer. *arXiv preprint arXiv:1912.11370*, 2019.

Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Joan Puigcerver, Jessica Yung, Sylvain Gelly, and Neil Houlsby. Big Transfer (BiT): General Visual Representation Learning. *arXiv:1912.11370 [cs]*, May 2020.

Simon Kornblith, Jonathon Shlens, and Quoc V Le. Do better imagenet models transfer better? In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2661–2671, 2019.

A. Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, Computer Science, University of Toronto, 2009.

Frederik Kunstner, Philipp Hennig, and Lukas Balles. Limitations of the empirical Fisher approximation for natural gradient descent. In *Advances in Neural Information Processing Systems*, pages 4156–4167, 2019.

Simon Laughlin. A simple coding procedure enhances a neuron's information capacity. *Zeitschrift für Naturforschung c*, 36(9-10):910–912, 1981.

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

Dong-Hyun Lee et al. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, page 896, 2013.

John Lee, Max Dabagia, Eva Dyer, and Christopher Rozell. Hierarchical optimal transport for multimodal distribution alignment. In *Advances in Neural Information Processing Systems*, pages 13474–13484, 2019.

David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

Hao Li, Pratik Chaudhari, Hao Yang, Michael Lam, Avinash Ravichandran, Rahul Bhotika, and Stefano Soatto. Rethinking the hyper-parameters for fine-tuning. In *Proc. of International Conference of Learning and Representations (ICLR)*, 2020.

Yingzhen Li and Richard E. Turner. R\'enyi Divergence Variational Inference. *arXiv:1602.02311 [cs, stat]*, October 2016.

Tengyuan Liang, Tomaso Poggio, Alexander Rakhlin, and James Stokes. Fisher-rao metric, geometry, and complexity of neural networks. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 888–896, 2019.

Chenghao Liu, Tao Lu, Doyen Sahoo, Yuan Fang, and Steven CH Hoi. Localized meta-learning: A PAC-Bayes analysis for meta-leanring beyond global prior. 2019.

Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. Deepfashion: Powering robust clothes recognition and retrieval with rich annotations. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018.

Stephan Mandt, Matthew D Hoffman, and David M Blei. Stochastic Gradient Descent as Approximate Bayesian Inference. *arXiv:1704.04289*, 2017.

Nathan Mantel. The detection of disease clustering and a generalized regression approach. *Cancer research*, 27(2 Part 1):209–220, 1967.

Jialin Mao, Itay Griniasty, Han Kheng Teoh, Rahul Ramesh, Rubing Yang, Mark K Transtrum, James P Sethna, and Pratik Chaudhari. The training process of many deep networks explores the same low-dimensional manifold. *arXiv preprint arXiv:2305.01604*, 2023.

Henry H Mattingly, Mark K Transtrum, Michael C Abbott, and Benjamin B Machta. Maximizing the information learned from finite data selects a simple model. *Proceedings of the National Academy of Sciences*, 115(8):1760–1765, 2018.

Andreas Mayer, Vijay Balasubramanian, Thierry Mora, and Aleksandra M Walczak. How a well-adapted immune system is organized. *Proceedings of the National Academy of Sciences*, 112(19): 5950–5955, 2015.

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

Robert J McCann. A convexity principle for interacting gases. *Advances in mathematics*, 128(1): 153–179, 1997.

Jameson Merkow, Robert Lufkin, Kim Nguyen, Stefano Soatto, Zhuowen Tu, and Andrea Vedaldi. DeepRadiologyNet: Radiologist level pathology detection in CT head images. *arXiv preprint arXiv:1711.09313*, 2017.

Marc Mezard and Andrea Montanari. *Information, physics, and computation.* Oxford University Press, 2009.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993, 2018.

Eric Nalisnick and Padhraic Smyth. Variational reference priors. 2017.

Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

Morteza Noshad, Yu Zeng, and Alfred O Hero. Scalable mutual information estimation using dependence graphs. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2962–2966. IEEE, 2019.

Barak A Pearlmutter. Fast exact multiplication by the hessian. *Neural computation*, 6(1):147–160, 1994.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

Anastasia Pentina and Christoph Lampert. A PAC-Bayesian bound for lifelong learning. In *International Conference on Machine Learning*, pages 991–999, 2014.

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.

Gabriel Peyré and Marco Cuturi. Computational Optimal Transport. *arXiv:1803.00567 [stat]*, April 2019.

Hang Qi, Matthew Brown, and David G Lowe. Low-shot learning with imprinted weights. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5822–5830, 2018.

Katherine N. Quinn, Colin B. Clement, Francesco De Bernardis, Michael D. Niemack, and James P. Sethna. Visualizing probabilistic models and data with intensive principal component analysis. *Proceedings of the National Academy of Sciences*, 116(28):13762–13767, 2019a. ISSN 0027-8424. doi: 10.1073/pnas.1817218116. URL https://www.pnas.org/content/116/28/13762.

Katherine N Quinn, Colin B Clement, Francesco De Bernardis, Michael D Niemack, and James P Sethna. Visualizing probabilistic models and data with intensive principal component analysis. *Proceedings of the National Academy of Sciences*, 116(28):13762–13767, 2019b.

Katherine N Quinn, Michael C Abbott, Mark K Transtrum, Benjamin B Machta, and James P Sethna. Information geometry for multiparameter models: New perspectives on the origin of simplicity. *Reports on Progress in Physics*, 2022.

C Radhakrishna Rao. Information and accuracy attainable in the estimation of statistical parameters. *Bulletin of the Calcutta Mathematical Society*, 37(3):81–91, 1945.

Rahul Ramesh and Pratik Chaudhari. Model zoo: A growing "brain" that learns continually, 2022a.

Rahul Ramesh and Pratik Chaudhari. Model Zoo: A Growing "Brain" That Learns Continually. In *Proc. of International Conference of Learning and Representations (ICLR)*, 2022b.

Rahul Ramesh, Jialin Mao, Itay Griniasty, Rubing Yang, Han Kheng Teoh, Mark Transtrum, James P Sethna, and Pratik Chaudhari. A picture of the space of typical learnable tasks. *arXiv preprint arXiv:2210.17011*, 2022.

CR Rao. Information and accuracy attainable in the estimation of statistical parameters. Kotz S & Johnson NL (eds.), Breakthroughs in Statistics Volume I: Foundations and Basic Theory, 235–248. 1945.

Ievgen Redko, Emilie Morvant, Amaury Habrard, Marc Sebban, and Younes Bennani. *Advances in domain adaptation theory*. Elsevier, 2019.

Maria Refinetti, Sebastian Goldt, Florent Krzakala, and Lenka Zdeborová. Classifying high-dimensional gaussian mixtures: Where kernel methods fail and neural networks succeed. In *International Conference on Machine Learning*, pages 8936–8947. PMLR, 2021.

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

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Mutual exclusivity loss for semi-supervised deep learning. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 1908–1912. IEEE, 2016.

Filippo Santambrogio. Optimal transport for applied mathematicians. *Birkäuser, NY*, 2015.

Andrew M Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D Tracey, and David D Cox. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, 2019.

James Sethna. *Statistical mechanics: entropy, order parameters, and complexity*, volume 14. Oxford University Press, 2006.

Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 806–813, 2014.

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

Eero P Simoncelli and Bruno A Olshausen. Natural image statistics and neural representation. *Annual review of neuroscience*, 24(1):1193–1216, 2001.

Joel G Smith. The information capacity of amplitude-and variance-constrained sclar Gaussian channels. *Information and control*, 18(3):203–219, 1971.

Alex J Smola and Bernhard Schölkopf. *Learning with kernels*, volume 4. Citeseer, 1998.

Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In

Advances in Neural Information Processing Systems, pages 4077–4087, 2017.

Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. FixMatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in Neural Information Processing Systems*, 33, 2020.

Jie Song, Yixin Chen, Jingwen Ye, Xinchao Wang, Chengchao Shen, Feng Mao, and Mingli Song. Depara: Deep attribution graph for deep knowledge transferability, 2020.

Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *arXiv preprint arXiv:1703.01780*, 2017.

Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *Proc. of the 37-Th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

Mark K Transtrum and Peng Qiu. Model reduction by manifold boundaries. *Physical review letters*, 113(9):098701, 2014.

Jesper E Van Engelen and Holger H Hoos. A survey on semi-supervised learning. *Machine learning*, 109(2):373–440, 2020.

Vladimir Vapnik. *Statistical learning theory.*, volume 3. Wiley, 1998.

Greg Ver Steeg and Aram Galstyan. Maximally informative hierarchical representations of high-dimensional data. In *Artificial Intelligence and Statistics*, pages 1004–1012, 2015.

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

Yi Xu, Lei Shang, Jinxing Ye, Qi Qian, Yu-Feng Li, Baigui Sun, Hao Li, and Rong Jin. Dash: Semi-supervised learning with dynamic thresholding. In *International Conference on Machine Learning*, pages 11525–11536. PMLR, 2021.

Rubing Yang, Jialin Mao, and Pratik Chaudhari. Does the data induce capacity control in deep learning? In *International Conference on Machine Learning*, pages 25166–25197. PMLR, 2022.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv:1605.07146*, 2016.

Amir R Zamir, Alexander Sax, William Shen, Leonidas J Guibas, Jitendra Malik, and Silvio Savarese. Taskonomy: Disentangling task transfer learning. In *Proceedings of the IEEE Con-*

ference on Computer Vision and Pattern Recognition, pages 3712–3722, 2018.

Chao Zhang, Lei Zhang, and Jieping Ye. Generalization bounds for domain adaptation. *Advances in neural information processing systems*, 4:3320, 2012.

Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. Mixup: Beyond empirical risk minimization. *arXiv:1710.09412*, 2017.

Zhongxin Zhang. *Discrete noninformative priors*. PhD thesis, Yale University, 1994.

Zhi-Hua Zhou and Ming Li. Semi-supervised learning by disagreement. *Knowledge and Information Systems*, 24(3):415–439, 2010.