Fine-tuning Generative Models

by

Arjun Khandelwal

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Masters of Engineering in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2019

Author . . . .
Department of Electrical Engineering and Computer Science
August 23, 2019

Certified by . . . .
David Sontag
Associate Professor
Thesis Supervisor

Accepted by . . . .
Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Fine-tuning Generative Models

by

Arjun Khandelwal

Submitted to the Department of Electrical Engineering and Computer Science
on August 23, 2019, in partial fulfillment of the
requirements for the degree of
Masters of Engineering in Computer Science and Engineering

Abstract

Deep generative models have emerged as a powerful modeling paradigm for making sense of large amounts of unlabeled real-world data. In particular, the representations produced by these models have proven to be useful both in improving human understanding of the factors of variation in the original dataset and in downstream tasks such as classification. Most current algorithms, however, require training a bespoke model from scratch, which can be both expensive and time-consuming. Instead, we propose various methods of fine-tuning pre-trained generative models to achieve these goals, and evaluate these methods quantitatively on few-shot classification and interpretability tasks.

Thesis Supervisor: David Sontag
Title: Associate Professor

Acknowledgments

First and foremost, thanks to David for being a truly excellent adviser. Among many other admirable traits, his infectious passion for improving the lives of others and care for his students have pushed me beyond what I thought capable. Perhaps most importantly, working with David has taught me the importance of asking good questions.

This work would not have been possible without Rahul G. Krishnan, who has spent countless hours guiding my research and entertaining my frequent questions. Thanks for being approachable, patient, and insightful throughout my two years in the lab, and for the constant (and vital) reminders to write.

There are many individuals who have made my journey at MIT as enjoyable as it has been. Thanks to my floormates in Burton Conner, friends in Phi Sig, and teammates on Grim for reminding me of the importance of life outside of work.

Lastly, thanks to my mother, Anu, for being a role model in all aspects of life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, the field of machine learning has seen a rapid rise in popularity due to a combination of algorithmic and computational advances. In particular, traditional *fully-supervised* algorithms have generated significant attention for their ability to match, and in some cases even exceed human performance on difficult image recognition and natural language processing tasks [9]. Unfortunately, this performance comes at a cost. Human-intensive and costly data-labeling efforts are necessary to produce labeled datasets of the requisite size to train state-of-the-art deep learning algorithms [18]. As the sheer amount of raw data grows exponentially due to the presence of smart appliances, industrial sensors, and countless other sources in a connected world, algorithms capable of utilizing unlabeled data are necessary.

The extent to which this plentiful unlabeled data is useful is not immediately clear, especially for problems that are of a discriminative nature, such as image classification. When sufficiently many labels are available, the incremental value of unlabeled data is negligible. Yet, as illustrated in the previous paragraph, labels can be scarce and generating them is expensive, especially when dealing with high-dimensional data. Recent work has shown that performance on tasks as wide-ranging as digit classification [15], recommendation systems [39], and object recognition [36] can be greatly improved with access to unlabeled examples (see Figure 1-1).

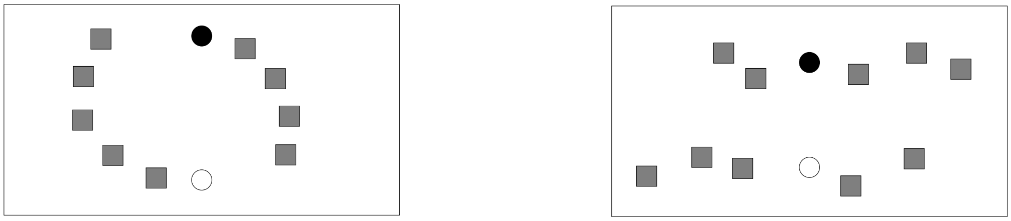

Figure 1-1: **Motivating Semi-supervised Learning:** Examples in which the unlabeled data illustrates which points are similar, and therefore influences the optimal decision boundary. In both plots, the colored circles represent labeled datapoints belonging to different classes, and the grey squares are unlabeled.

A common means of leveraging unlabeled data is via *representation learning*, which seeks lower-dimensional embeddings of the data in which some notion of distance corresponds to similarity in the original space. Once found, these may be used in a variety of ways across many downstream tasks. For example, a predictor may be trained on the embeddings of the labeled subset, resulting in a classifier less likely to overfit. Alternatively, the embeddings themselves may be used to visualize and identify factors of variation within high-dimensional datasets for the purposes of human interpretability. A variety of algorithms to learn these embeddings have been proposed, including linear techniques such as PCA and factor rotation [27] and their non-linear variants such as t-SNE [23].

In this thesis, we consider the *deep generative model*, a Bayesian, probabilistic approach to representation learning. Deep generative models attempt to approximate the data distribution via a pre-specified prior over a low-dimensional latent variable z along with a neural network-parameterized conditional density $p(x|z)$. Although computation of the exact posterior $p(z|x)$ is intractable, tools from variational inference and stochastic gradient estimation may be used to learn an approximation $q(z|x)$. The individual dimensions of z may then be interpreted as nonlinear factors, with the approximate posterior $q(z|x)$ usable as a low-dimensional embedding of x. Deep generative models have multiple uses which include density estimation, dimensionality reduction, and semi-supervised learning, and have displayed remarkable results even when applied to complex, high-dimensional datasets.

Treated solely as a maximum-likelihood estimation problem, the generating distribution $p(x|z)$ is unidentifiable; that is, there are infinitely many choices of $p(x|z)$ which result in the same marginal density $p(x)$, and therefore infinitely many possible representations. The use of various inductive biases based on varying combinations of both labeled and unlabeled data to guide the learned representations towards capturing relationships useful for classification or human-interpretability is therefore of interest. When beginning with a trained deep generative model, rather than training the parameters of a bespoke generative model jointly, we refer to this process as *fine-tuning.*

An appealing property of fine-tuning-based approaches in the context of real-world problems is that a generative model trained only once may be copied and adapted to a variety of tasks. As as a result, the computationally intensive training process and large set of unlabeled examples necessary for training good generative models for complex datasets need only be present in one place, with the parameters of the resulting model shared at will. This is especially relevant when sufficient quantities of unlabeled data are expensive or even impossible to obtain, as in the case of personal health data.

In this thesis, we study the problem of fine-tuning deep generative models for use on various downstream tasks. In Chapters 3 and 4, we present new approaches to the semi-supervised few-shot learning problem which leverage a generative model. In Chapter 5, we address the fully unsupervised setting, and propose a method for increasing the disentanglement of the learned latent space of a given trained generative model.

1.1 Contributions

Improving Latent Reasoning Networks: The Latent Reasoning Network, proposed in [17], introduces a neural architecture and learning algorithm capable of computing the similarity between a query set and an object. Drawing inspiration from few-shot learning, we propose modifications to the loss function and computation of the Bayes factor that result in significant performance improvements on a benchmark few-shot learning task.

Generative Comparison Networks: A New Method for Few-Shot Learning:

Extending purely discriminative techniques for fully-supervised few-shot learning, we introduce the Generative Comparison Network, which evaluates the similarity between datapoints via comparison of their posterior distributions in latent space under a fine-tuned generative model. We propose a variety of suitable measures of similarity, and evaluate them individually against a challenging semi-supervised few-shot learning task, displaying competitive performance.

Disentangled Representations via Decorrelation: We introduce a new method for disentanglement involving an *ex post* modification of a learned generative model. In contrast to existing methods such as the DIP-VAE [19] and TC-VAE [2], ours is computationally efficient, requires no hyper-parameter tuning, and provably improves the unmodified generative model. We evaluate on a variety of different supervised and supervised disentanglement metrics, showing competitive performance while avoiding some of the tradeoffs present in existing methods.

Chapter 2

Background

2.1 Variational Autoencoders

An important building block in our model is the Variational Autoencoder (VAE), a commonly used deep generative model. Given observed data $x \in \mathbb{R}^d$ generated from an unknown distribution $p(x)$, and latent variable vector $z \in \mathbb{R}^l$ with prespecified prior $p(z)$, the VAE consists of an *inference network* $q_\phi(z|x)$ and *recognition network* $p_\psi(x|z)$, parameterized by neural networks. Here, $q_\phi(z|x)$ is a variational approximation to the true posterior $p(z|x)$. In most cases, we will use a Gaussian VAE, where $q_\phi(z|x) = N(\mu_\phi(x), \Sigma_\phi(x))$ and $p(z) = N(\mathbf{0}, I_l)$.

Suppose we have access to k unlabeled samples $x_1, x_2, ..., x_k \sim p(x)$. Training the VAE proceeds by stochastic gradient ascent on the *evidence lower bound* (ELBO); a tractable lower bound for the data log-likelihood $\sum_{i=1}^k \log p(x_i)$, defined as follows:

$$\mathcal{L}^{ELBO} = \sum_{i=1}^{k} \left(\mathbb{E}_{z \sim q_\phi(z|x_i)}[\log p_\psi(x_i|z)] - KL(q_\phi(z|x_i)||p(z)) \right) \tag{2.1}$$

For our purposes, it will be helpful to decompose $\mathcal{L}^{ELBO}$ as the sum of the *recon-*

struction loss $\mathcal{L}^R$ and the *KL loss* $\mathcal{L}^{KL}$, defined as follows:

$$\mathcal{L}^R = \sum_{i=1}^{k} \left(\mathbb{E}_{z\sim q_\phi(z|x_i)}[\log p_\psi(x_i|z)]\right)$$
$$\mathcal{L}^{KL} = -\sum_{i=1}^{k} \left(KL(q_\phi(z|x_i)||p(z))\right)$$

To compute noisy gradients of $\mathcal{L}^{ELBO}$ with respect to the parameters of the recognition network and inference network, the *reparameterization trick* is employed ([14], [31]).

Since their introduction, VAEs have been the subject of many papers seeking to understand their properties or proposing extensions. While we primarily use the "vanilla" VAE described above for the sake of simplicity of implementation and fair comparison to existing benchmarks, our methods may be applied without much difficulty to many of the more complex models that have since been developed.

2.2 Few-Shot Learning

To motivate the problem of few-shot learning, consider the following observation: a toddler presented with their very first image of a giraffe is able to identify giraffes in other images (regardless of size, position, or surrounding content) without much difficulty. This ability stands in stark contrast to standard machine learning methods in image classification, which require hundreds of labeled images in order to reliably identify a given class. In other words, given an understanding of other, similar classes of objects (in this case, other animals), a human is able to obtain an accurate understanding of a new class from very few labeled examples. The process by which this occurs is commonly referred to as *few-shot learning.*

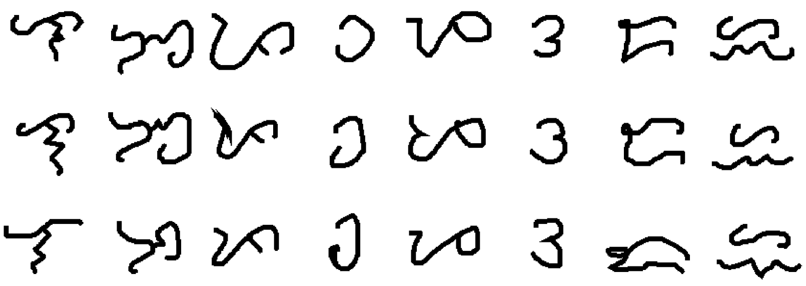

Figure 2-1: **Omniglot Visualization:** Image samples from the Omniglot dataset are shown, with each column representing a distinct class.

In this thesis, we consider the question of how to develop algorithms to replicate this ability. To do so, we must first introduce a task which serves as an evaluation of few-shot learning capability.

We define the n-way, k-shot classification task as follows: during training, we are given a labeled dataset

$$\mathcal{D}^L := \{(x_1, y_1), (x_2, y_2), ..., (x_m, y_m)\}$$

with the labels $y_i \in \{1, 2, \ldots, c\}$ and an unlabeled dataset

$$\mathcal{D}^U := \{x_{m+1}, x_{m+2}, ..., x_{m+m'}\}$$

For each test time *instance* a labeled *support set*

$$\mathcal{S} := \{x_1^{(1)}, x_2^{(1)}, \ldots, x_k^{(1)}, x_1^{(2)}, \ldots x_k^{(n)}\}$$

with n classes and k datapoints per class is given, and we are tasked with classifying a query point x as one of the n classes. Our performance is the accuracy averaged across many such instances. Unlike the traditional classification setting, the n classes present in $\mathcal{S}$ do not overlap with the classes observed during training time. To simplify our presentation with this in mind, we refer to these support set classes as $1 \dots n$. Fully-supervised few-shot learning and its semi-supervised variant are distinguished by whether $\mathcal{D}^U$ is empty.

As is standard in the literature, we evaluate our proposed algorithms on the Omniglot dataset (see Figure 2-1), introduced in [20]; a collection of handwritten characters created specifically for the few-shot learning setting. This dataset consists of 20 grayscale iamges for each of 1623 characters drawn from a total of 50 different alphabets, resized from 105x105 to 28x28. Following [38], we split the dataset into 1200 training classes and 423 evaluation classes. Support sets for the test-time instances are sampled randomly from the latter. In the semi-supervised setting, 90% of samples from each training class chosen at random are moved to $\mathcal{D}^U$.

A variety of approaches to few-shot learning have been proposed. These may be broadly classified as follows:

Metric Learning: Few-shot learning is closely related to the problem of metric learning, which seeks a means of measuring similarity between objects. [38], [34] and [35] all accomplish this by learning a map from the data to a lower-dimensional embedding space, parameterized by a deep neural network. The similarity used in each is different, with the authors using the cosine similarity, Euclidean distance, and a jointly learned function respectively.

Meta-learning: Building on the metric-learning literature, meta-learning approaches seek to improve generalization to unseen data by learning ways to modify their prediction method during prediction, using the data provided during testing. [28] and [26] use an auxiliary neural network to update model weights at test-time, where [6] use test-time gradient updates to fine-tune their model for prediction.

Utilizing Generative Models: [10] and [5] use learned generative models for few-shot learning. In both works, the authors use an explicitly parameterized context variable as part of the generative process. For few-shot prediction, posterior distributions over the context variables are compared for the query and the classes in the support set.

Semi-supervised Few-shot Learning: Recent work has considered the extension of few-shot learning to settings where portions of the training or test data are unlabeled. Of these, [29] augments the standard prototypical networks algorithm with an embedding refining procedure making use of unlabeled data during test-time. [41] use samples from a GAN trained partly on the unsupervised data to fine-tune their discriminative decision boundaries.

In particular, our algorithms will draw inspiration from previous work from metric learning and utilizing generative models to allow few-shot learning in both the fully-supervised and semi-supervised setting.

Chapter 3

Improving the Latent Reasoning Network

In this chapter, we refine the approach taken by [17] to few-shot learning, which *fine-tunes* a pre-existing generative model using label information. We suggest a number of improvements to both the evaluation and learning stages of the previous framework.

For *evaluation*, we derive a closed form of the approximation to the Bayes factor in the case where the the prior and variational posteriors lie in the same exponential family. This removes the need for Monte Carlo sampling during evaluation. For *learning*, we identify multiple shortcomings of the original max-margin loss function in the context of classification and suggest means around them.

The effects of these changes are demonstrated via an ablation study, showing significant performance improvements on a benchmark few-shot learning task. With the improved algorithm we demonstrate competitive results on a difficult semi-supervised learning task.

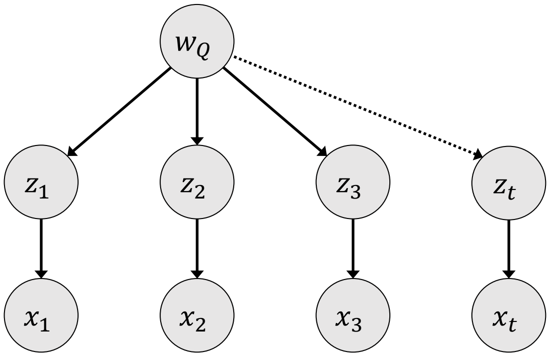

Figure 3-1: **Latent Variable Model for the Latent Reasoning Network:** A latent variable w_Q determines the shared commonality between objects in a set. Based on the value of w_Q, the per-datapoint latent variable z is generated. The dashed arrow represents the hypothesis that the query x_t was generated from the same realization of w_Q.

3.1 The Latent Reasoning Network

In this section, we describe the Latent Reasoning Network (LRN), a method for reasoning about the similarity between a query x_t and a set of objects $Q := \{x_1, x_2, \ldots, x_Q\}$ introduced in [17]. To answer such queries, the LRN assumes the latent variable model depicted in Figure 3-1. In this model, w represents a shared trait of objects in the set. For example, this might be class identity in few-shot learning or a particular user in recommender systems. Based on this individual property, a per-datapoint latent variable $z \in \mathbb{R}^l$ is generated, which then gives rise to the observed features.

To evaluate the similarity between an object x and a set Q given such a generative model, the model makes use of the *Bayes Factor*, a normalized score defined as

$$\text{score}(x, Q) = \log \frac{p(x|Q)}{p(x)}$$

The Bayes factor has a rich history as a scoring function, first introduced in the context of document retrieval by [7]. One means of motivating the Bayes factor is via hypothesis testing. In particular, the Bayes factor is the likelihood ratio arising from the hypothesis that x the observations in Q were generated with the same value of w, implying high similarity, as compared to the null hypothesis that x was independently generated.

Exact computation of both the posterior density $p(x|Q)$ and marginal density $p(x)$ is intractable. Accordingly, the authors of [17] rewrite the Bayes factor as follows using Bayes rule along with conditional independence relationships:

$$\log \mathrm{score}(x, Q) = \log \int \frac{p(z|Q)p(z|x)}{p(z)} dz \tag{3.1}$$

The authors then propose to approximate each of the terms appearing in Equation 3.1 separately, with the integral approximated via importance sampling. $p(z|x)$ is replaced with $q_\phi(z|x)$, the inference network of a Gaussian variational autoencoder with parameters ϕ and recognition network $p_\psi(x|z)$, referred to as the *data model*. $p(z)$ is a standard normal Gaussian. Lastly, $p_\theta(z|Q)$ is amortized by means of a *reasoning model*, which is a learned map parameterized by θ. The reasoning model takes as input the individual Gaussian posteriors $\{q_\phi(z|x_i)\}_{1\leq i\leq Q}$, computed using the inference network of the data model, and outputs a single Gaussian distribution over the latent space. To accomodate both the permutation-invariant set structure and variable size of the input, the reasoning model uses the DeepSets architecture specifically designed with these properties from [40].

Put together, these approximations yield

$$\log \mathrm{score}(x, Q) \approx \log \mathbb{E}_{q_\phi(z|x)} \left[\frac{p_\theta(z|Q)}{p(z)} \right] \tag{3.2}$$

In the following sections, we refer to the right-hand side as $\log \mathrm{score}(x, Q)$.

3.2 Learning Algorithm for the Latent Reasoning Network, from [17]

Before discussing our modifications, we review the *loss function* and the *learning algorithm* presented in [17]. In broad terms, the goals of training are two-fold: first, to fine-tune the latent space of the data model towards the task at hand; and second, to learn the parameters of the reasoning model. We assume that we are in the few-shot classification setting as defined in Section 2.2. As a result, we are presented with labeled data at training time. To accomplish the objectives in the context of a classification task, the authors propose a *max-margin* discriminative loss function based on the intuition that given a query set with a single shared label among elements, objects with the same label should have a higher score than those that do not.

To begin, we make use of the following definitions: for arbitrary $(x, y) \in \mathcal{D}^L$,

$$X_s := \{x' | (x', y') \in \mathcal{D}^L, y = y'\} \tag{3.3}$$

$$X_{ns} := \{x' | (x', y') \in \mathcal{D}^L, y \neq y'\} \tag{3.4}$$

Then, for a datapoint x, the max-margin loss $\mathcal{L}^{mm}(x, \phi, \psi)$ is defined as follows:

$$\begin{aligned} \mathcal{L}^{mm}(x, \phi, \theta) = {} & \mathbb{E}_{Q_s \subset X_s} \mathbb{E}_{Q_{ns} \subset X_{ns}} \\ & \left[\frac{1}{d} \sum_{x_{ns} \in Q_{ns}} \max \left(\log \text{score}(x_{ns}, Q_s) - \log \text{score}(x, Q_s) + \Delta, 0 \right) \right] \end{aligned} \tag{3.5}$$

where the size of Q_s and Q_{ns} is the hyperparameter b, and Δ is the margin, taken to be the squared distance between the posterior means of x and x_{ns}. As a natural form of regularization, ensuring that the outputs of the data model still overlap with the prior and capture variation in the data, we include $\mathcal{L}^{ELBO}(x, \phi, \psi)$ as an unsupervised term in the full objective. $\mathcal{L}^{mm}$ and $\mathcal{L}^{ELBO}$ are weighted according to a hyperparameter C, giving the combined loss:

$$\mathcal{L}^{C}(\phi, \psi, \theta) = \mathbb{E}_x \left[\frac{C}{C+1} \mathcal{L}^{mm}(x, \phi, \theta) + \frac{1}{C+1} \mathcal{L}^{ELBO}(x, \phi, \psi) \right] \tag{3.6}$$

To begin training, the data model is initialized as a variational autoencoder pretrained on all available datapoints, unlabeled or labeled. Following this step, gradient descent is performed on $\mathcal{L}^C$ for the parameters ϕ, ψ, and θ, jointly learning the reasoning model and fine-tuning the latent space via backpropagation through the inference/recognition networks.

To apply the Latent Reasoning Network to an individual few shot classification instance, as defined in Section 2.2, for $1 \leq i \leq n$ we define $Q^{(i)} := \{x_1^{(i)}, x_2^{(i)}, \ldots x_k^{(i)}\}$. As $Q^{(i)}$ is the subset of the support set corresponding to the single class i, we label the query x as $\arg\max_i \text{score}(x, Q^{(i)})$. Crucially, the reasoning model permits a variable number of inputs by construction, permitting usage of a single model for all k.

3.3 Evaluating the Latent Variable Bayes Factor

In this section, we show that the Latent Variable Bayes factor may be computed analytically, avoiding the variance associated with the Monte Carlo-based sampling approach used in [17]. Our result is stated generally as follows:

Proposition 1. *Suppose $q(z|x), q(z|Q), p(z)$ have density functions of the form*

$$h(z) \exp(\langle \theta, T(z) \rangle - A(\theta))$$

with parameters $\theta_x, \theta_Q, \theta_p$ respectively. Then,

$$\int_z \frac{q(z|x)}{p(z)} q(z|Q)dz = A(\theta_x + \theta_Q - \theta_p) - (A(\theta_x) + A(\theta_Q) - A(\theta_p))$$

Proof.

$$\begin{aligned}
&\int_z \frac{q(z|x)}{p(z)} q(z|Q)dz \\
&\qquad = \int_z \frac{h(z)\exp(\langle\theta_x, T(z)\rangle - A(\theta_x))}{h(z)\exp(\langle\theta_p, T(z)\rangle - A(\theta_p))} h(z)\exp(\langle\theta_Q, T(z)\rangle - A(\theta_Q))dz \\
&\qquad = \int_z \frac{\exp(\langle\theta_x, T(z)\rangle - A(\theta_x) + \langle\theta_Q, T(z)\rangle - A(\theta_Q))}{\exp(\langle\theta_p, T(z)\rangle - A(\theta_p))} h(z)dz \\
&\qquad = \int_z h(z)\frac{\exp(\langle\theta_x, T(z)\rangle + \langle\theta_Q, T(z)\rangle - \langle\theta_p, T(z)\rangle}{\exp(A(\theta_x) + A(\theta_Q) - A(\theta_p))} dz \\
&\qquad = \frac{1}{\exp(A(\theta_x) + A(\theta_Q) - A(\theta_p))} \int_z h(z)\exp(\langle\theta_x + \theta_Q - \theta_p, T(z)\rangle)dz \\
&\qquad = \frac{\exp(A(\theta_x + \theta_Q - \theta_p))}{\exp(A(\theta_x) + A(\theta_Q) - A(\theta_p))}
\end{aligned}$$

□

The analytic form given by Proposition 1 holds for all cases in which the variational approximations $q(z|Q), q(z|x)$, and the prior $p(z)$ are members of the same exponential family. When these are Gaussians with diagonal covariance matrices, as is the case for the Latent Reasoning Network, the following corollary holds:

Corollary 1. *Suppose $q(z|x), q(z|Q)$, and $p(z)$ are $N(\mu_x, \Sigma_x), N(\mu_Q, \Sigma_Q)$, and $N(0, I_k)$, respectively.*

Then, for $1 \le i \le k$, letting $A_i = (\Sigma_x)_{ii} + (\Sigma_q)_{ii} - (\Sigma_x)_{ii}(\Sigma_q)_{ii}$,

$$\begin{aligned}
&\log \int_z \frac{q(z|x)}{p(z)} q(z|Q)dz \\
&\qquad = \sum_{i=1}^{k} \left(\frac{-((\mu_x)_i - (\mu_Q)_i)^2 + (\mu_x)_i^2(\Sigma_Q)_{ii} + (\mu_Q)_i^2(\Sigma_x)_{ii}}{2A_i} - \log\sqrt{A_i} \right)
\end{aligned}$$

To derive this statement, each of $q(z|x), q(z|Q)$, and $p(x)$ is written in the exponential family form, then Proposition 1 is applied. The closed-form expression for $\text{score}(x, Q)$ given by Corollary 1 may be used when computing the original loss defined in Equation 3.6, the new losses introduced in Section 3.4, and the few-shot classification procedure described in Section 3.2. Accordingly, both training and evaluation of the Latent Reasoning Network are affected.

The Latent Variable Bayes Factor and its closed form for exponential families may be of interest in the broader context of determining similarity between probability distributions. In particular, the LVBF is a kernel; a property that is apparent from its integral form. Therefore the LVBF may be used an alternative to standard choices for kernel-based learning algorithms such as SVMs classifying distributions, rather than points [12].

3.4 Improving the Latent Reasoning Network

In this section, we describe the proposed modifications to the Latent Reasoning Network. While we leave the structure of the model constant, as described in Section 3.1, we will introduce a variety of changes to the loss function with the shared motivation of aligning the training-time loss with the test-time few-shot classification task.

3.4.1 Adding a Classification-Based Objective

Within a single few-shot classification instance the query point is shared among the score evaluations, while the set of objects is varied among those consisting of a single class. This stands in contrast to $\mathcal{L}^{mm}$ as defined in Equation 3.5, where the set Q is held constant while the query point varies. This observation motivates the introduciton of a new loss function, defined as follows:

$$\mathcal{L}_2^{mm}(x,\phi,\theta) = \mathbb{E}_{Q_s \subset X_s}\mathbb{E}_{Q_{ns} \subset X_{ns}}\left[\max\left(\log\ \text{score}(x,Q_s) - \log\ \text{score}(x,Q_{ns}),0\right)\right] \tag{3.7}$$

While it is possible to fully replace $\mathcal{L}^{mm}$ with $\mathcal{L}_2^{mm}$ in the combined loss $\mathcal{L}^C$, we find from experiments on the Omniglot dataset that incorporating both terms with equal weights performs best. In particular, our new combined loss function is

$$\mathcal{L}_1^C(\phi,\psi,\theta) = \mathbb{E}_x\left[\frac{C}{C+1}\frac{\mathcal{L}^{mm}(x,\phi,\theta)+\mathcal{L}_2^{mm}(x,\phi,\theta)}{2} + \frac{1}{C+1}\mathcal{L}^{ELBO}(x,\phi,\psi)\right] \tag{3.8}$$

Note that $\mathcal{L}_2^{mm}$ may be computed solely using the sets of datapoints sampled for $\mathcal{L}^{mm}$, bringing the total additional computation requirement during training to a single forward/backwards pass for $\log \text{score}(x, Q_{ns})$ for each element of a batch.

3.4.2 Improving Separation using a Log-Loss

Next, we observe that once $\log \text{score}(x,Q_s) - \log \text{score}(x,Q_{ns}) > \Delta$, the gradient of $\mathcal{L}^{mm}$ is 0. As a result, for small values of Δ in particular, the Latent Reasoning Network may lack the ability to "push apart" datapoints from different classes enough to guarantee generalizable class separation. To remedy this, we suggest replacing $\mathcal{L}^{mm}$ and $\mathcal{L}_2^{mm}$ with $\mathcal{L}^{log}$ and $\mathcal{L}_2^{log}$, defined as follows:

$$\begin{aligned}\mathcal{L}^{log}(x,\phi,\theta) = {} & \mathbb{E}_{Q_s \subset X_s}\mathbb{E}_{Q_{ns} \subset X_{ns}} \\ & \left[\frac{1}{b}\sum_{x_{ns} \in Q_{ns}} -\log \frac{\text{score}(x,Q_s)}{\text{score}(x,Q_s)+\text{score}(x_{ns},Q_s)}\right]\end{aligned} \tag{3.9}$$

$$\mathcal{L}_2^{log}(x, \phi, \theta) = \mathbb{E}_{Q_s \subset X_s} \mathbb{E}_{Q_{ns} \subset X_{ns}} \left[-\log \frac{\text{score}(x, Q_s)}{\text{score}(x, Q_s) + \text{score}(x, Q_{ns})} \right] \tag{3.10}$$

In doing so, the "weight" of datapoint x within the batch gradient computation for $\mathcal{L}_2^{log}$ is inversely related with the size of $\text{score}(x, Q_s)$ relative to $\text{score}(x, Q_{ns})$ while still being non-negative (and the analogous statement holds for $\mathcal{L}^{log}$). This is desired behavior, as during training, parameter updates which distinguish hard-to-separate classes should be prioritized. In contrast, the weight of datapoint x when taking gradients of $\mathcal{L}^{mm}$ and $\mathcal{L}_2^{mm}$ does not depend on the relative values of $\text{score}(x, Q_s)$ and $\text{score}(x, Q_{ns})$, until the point at which $\log \text{score}(x, Q_s) - \log \text{score}(x, Q_{ns}) > \Delta$, when it falls abruptly to 0.

For further intuition regarding the new loss terms, suppose that in all cases the set Q consists solely of from a single class c. If $\text{score}(x, Q) \propto p(x \in \text{class } c)$, then $\mathcal{L}_2^{log}$ is none other than the log-loss for a binary classification task (identifying all images with a different label than x as the same class), with a similar statement holding for $\mathcal{L}^{log}$. In using the loss from this binary classification task during training to approach the few-shot classification task, we recover the approach of Siamese networks for few-shot learning, introduced in [16].

The intuition behind our new objectives relies on our observed similarities during training being a class partition of the dataset, rather than e.g. real-valued similarity judgements between objects and sets. This illustrates another way in which our modifications leverage the particular structure of the classification problem.

The updated loss function is:

$$\mathcal{L}_2^C(\phi,\psi,\theta) = \mathbb{E}_x\left[\frac{C}{C+1}\frac{\mathcal{L}^{log}(x,\phi,\theta)+\mathcal{L}_2^{log}(x,\phi,\theta)}{2} + \frac{1}{C+1}\mathcal{L}^{ELBO}(x,\phi,\psi)\right] \quad (3.11)$$

3.5 Number of Training Ways

As defined, $\mathcal{L}_2^{log}(x,\phi,\theta)$ corresponds to a binary classification question, as described in the previous section. Few-shot classification, however, is a multi-way classification task when the number of ways is larger than two, and so the final modification we propose will refine $\mathcal{L}^{log}(x,\phi,\theta)$ and $\mathcal{L}_2^{log}(x,\phi,\theta)$ to enable comparison against multiple classes.

Recall from Section 2.2 that our training-time labels are $\{1\dots c\}$. We specify t as a hyperparameter controlling the number of "training ways". For a given $(x,y) \in \mathcal{D}^L$ and specified t, we let $S \subset \{1\dots l\} \setminus \{y(x)\}, |S|= t$ be a subset of t distinct class labels chosen at random. For $1 \leq i \leq t$, we sample $Q_{ns}^{(i)} \subset \{x'|(x',y') \in \mathcal{D}^L, y' = S_i\}, |Q_{ns}^{(i)}|= b$. The final loss functions are defined as follows:

$$\begin{aligned}\mathcal{L}_3^{log}(x,\phi,\theta) = \mathbb{E}_{Q_s \subset X_s}\mathbb{E}_S\mathbb{E}_{Q_{ns}^{(i)}} \\ \left[\frac{1}{b}\sum_{j=1}^{b} -\log\frac{\text{score}(x,Q_s)}{\text{score}(x,Q_s)+\sum_{i=1}^{t}\text{score}((Q_{ns}^{(i)})_j,Q_s)}\right]\end{aligned} \quad (3.12)$$

$$\begin{aligned}\mathcal{L}_4^{log}(x,\phi,\theta) = \mathbb{E}_{Q_s \subset X_s}\mathbb{E}_S\mathbb{E}_{Q_{ns}^{(i)}} \\ \left[-\log\frac{\text{score}(x,Q_s)}{\text{score}(x,Q_s)+\sum_{i=1}^{t}\text{score}(x,(Q_{ns}^{(i)})_j)}\right]\end{aligned} \quad (3.13)$$

In order to replicate the test-time n-way classification setting, we would set $t = n-1$. As observed by [34], models trained with slightly larger t typically perform better.

For informal intuition, we suppose t is small. In this case, the gradient update for both $\mathcal{L}_3^{log}$ and $\mathcal{L}_4^{log}$ might unwittingly push the representation of x in the latent space towards those of a class not seen in any of the $Q_{ns}^{(i)}$, compromising separation and therefore classification accuracy. As t increases however, the latent representation of x is pushed away from many classes at once, resulting in a more "robust" separation. This improvement comes at a cost; the computational requirement of a single gradient computation increases linearly in t. We take $t = 29$ in our experiments, as we find that increasing t beyond this point results in negligible performance gains.

The final loss function is:

$$\mathcal{L}_3^C(\phi, \psi, \theta) = \mathbb{E}_x \left[\frac{C}{C+1} \frac{\mathcal{L}^{log}(x, \phi, \theta) + \mathcal{L}_2^{log}(x, \phi, \theta)}{2} + \frac{1}{C+1} \mathcal{L}^{ELBO}(x, \phi, \psi) \right] \quad (3.14)$$

3.6 Experiments and Results

We evaluate our algorithm in a variety of few-shot classification setups using the standard Omniglot dataset. We follow standard data augmentation procedures at training time. These include augmenting the original classes by applying random small rotations, shifts, and scalings, and generating new training classes by rotating images in the original classes through multiples of 90°. For all experiments, we take $t = 29$. Increasing t beyond this point substantially increases training time while providing no clear increase in out-of-sample accuracy. During both pretraining of the data model VAE and training of the LRN, we use the Adam optimizer with a learning rate of 0.0003. For pretraining, we use a batch size of 256 images for 500 epochs, and for training of the LRN we use a batch size of 32 instances for 8 epochs. For the semi-supervised setting, we find that taking $C = 250$ strikes the right balance between the discriminative loss regularization of the generative loss, while for the fully-supervised setting using solely the discriminative term (i.e. letting $C = \infty$) performs best.

For Omniglot, the data model is a VAE with convolutional encoder and deconvolutional decoder architectures identical to those used in [5]. Compared to [5], however, we use a simpler generative model with no skip-connections and only one set of latent variables. The reasoning model consists of one permutation-equivariant layer. We use the training algorithm described above.

3.6.1 Ablation Study

To further understand the individual significance of the modifications to the learning procedure, we examine performance in the fully supervised setting with each removed. The results, displayed in Table 3.1, show that each individual modification plays a non-negligible role in the final performance evaluation. In particular, for each instance increasing the number of classes the datapoint is compared against (as suggested in Section 3.5) plays an important role. We were unable to test the incremental value of removing the max-margin, as there is no clear analogue of the max-margin loss for $t > 1$.

Table 3.1: Accuracy on the 5-way Fully-supervised Omniglot task

Model	1-shot	5-shot
Ours, Equation 3.14 (Final)	98.0	99.3
[17], Equation 3.6 (Original Max-Margin Loss)	93.6	98.2
Ours, Equation 3.11 (Binary Comparisons)	96.7	99.0
Ours, Equation 3.14* ($\mathcal{L}_2^{log}$ Removed)	97.5	99.1

3.6.2 Fully-supervised Few-shot Learning

At evaluation time in the fully-supervised n-way k-shot setting, for an individual task T_i we randomly choose n classes from the evaluation set and sample k images from each of the classes, which together constitute the *support set*. One of the n classes is selected at random, and an additional *exemplar image* is drawn from this class. The reported performance is the model's accuracy in identifying the class from which the exemplar image is drawn, averaged across many individual tasks T_i.

Table 3.2: Accuracy on the 5-way Omniglot task

Model	1-shot	5-shot
Ours	98.0	99.3
LRN (UAI 2018) [17]	93.6	98.2
Matching Networks [38]	98.1	98.9
Neural Statisician [5]	98.1	99.5
Prototypical Nets [34]	98.8	99.7
Relation Nets	99.6	99.8
MetaGAN [41]	99.7	99.9

In the fully-supervised setting we observe that with the changes of Section 3.3 and Section 3.4, our method is competitive with similar algorithms making use of deep generative models (e.g. [5]). Of note is the remarkable performance of the Relation Network and its MetaGAN variant. Instead of pre-specifying a similarity function, such as the Euclidean distance, the Relation Net algorithm learns a similarity function parameterized by a neural network jointly with the embeddings. When sufficient labeled data is available to learn the increased parameter set, the added flexibility afforded by the learned similarity results in better generalization. We discuss analogous potential methods to increase the flexibility of the learned latent space of the LRN in Section 3.7.

3.6.3 Semi-supervised Few-shot Learning

The semi-supervised n-way k-shot setting we consider mirrors that of [29] and [41] at *training time*. In this context, for each training class 10% of images are randomly chosen to keep their labels, while the remainder constitute the unlabeled set. At evaluation time, the individual tasks T_I are constructed in an identical manner to the fully-supervised task. This differs from the setup used in [29], which, in addition to the labeled data in the support set uses previously unseen unlabeled data at evaluation time, and thus uses strictly more information. This setup is referred to as *Task-Level Semi-Supervised Few-Shot Learning* in [41].

Table 3.3: Accuracy on the 5-way Semi-supervised Omniglot task

Model	1-shot
Ours	96.8
LRN [17]	92.4
MetaGAN [41]	97.1
Relation Net [35]	93.8
Prototypical Nets [34]	93.7

We show the results of the semi-supervised evaluation in Table 3.3. Our method is competitive with the state-of-the-art MetaGAN, and considerably outperforms baselines solely making use of labeled data. The results support fine-tuning generative models as a means to accomplish few-shot learning in a label- and data-efficient manner.

3.7 Discussion

Building on the work of [17], we introduced multiple improvements to the learning procedure for the Latent Reasoning Network and quantified their contribution through an ablation study on the Omniglot few-shot learning task. Replacing the max-margin loss with the log-loss and increasing the number of classes in a training instance qualitatively results in a latent space fine-tuned towards pushing apart points in different classes while pulling together points in the same class. We also add an additional training-time loss term inducing the score between a fixed target point and a query set to be highest when the query set consists of objects from the same class as the target point. Doing so closes the gap between training and evaluation objectives yielding quantifiable improvements in test-time accuracy.

When all the terms in the approximation to the Bayes Factor lie in the exponential family, we derive an analytic expression for the Latent Variable Bayes Factor. Doing so yields a lower variance replacement to the Monte Carlo estimator previously used in both the evaluation procedure and training-time gradient estimates.

The benefits of our changes are evident in both fully-supervised and semi-supervised few-shot learning tasks. For the latter we show competitive results with state-of-the-art models, making a case for the use of unsupervised pretraining followed by discriminative fine-tuning.

A limitation of the current model is the use of the entire latent space in evaluating similarity. This enforces an implicit tradeoff between discriminative and generative performance, which we suspect is the driver behind the decrease in quality of generated images after discriminative training. Adding auxiliary latent variables to model per-datapoint variability in attributes beyond class identity is a possible workaround.

Another possible extension is exploring the use of alternatives to the standard Gaussian VAE for the data model. Either an increased ability to model variability among individual datapoints or a means of learning latent spaces in which inter-class and intra-class variation are more easily distinguished could translate to more refined similary judgements, resulting in better performance on downstream tasks (such as few-shot learning). The former may be achieved by using more flexible prior ([37]) or posterior ([30]) parameterizations. For the latter, we might for example learn a latent space consisting jointly of both discrete and continuous representations, as in [3]. For particular choices of discrete random variables (e.g. Bernoulli, Poisson) the closed-form score presented in Section 3.3 will still be usable.

Chapter 4

Generative Comparison Networks

In this chapter, we introduce the *Generative Comparative Network*, a classification algorithm designed for a semi-supervised setting in which a generative model is the natural means of utilizing the information from unlabeled data. Along the way, we introduce efficiently computable similarity measures between datapoints based on their corresponding posteriors in latent space. We present a semi-supervised learning algorithm which leverages these similarity measures and class information to *fine-tune* the latent space of an existing generative model using a loss function and training algorithm inspired by few-shot learning. We relate our method to previous algorithms, such as Matching Networks, Prototypical Networks, and the Latent Reasoning Network introduced in Chapter 3, and display its potential on a challenging semi-supervised few-shot learning task, on which the GCN performs nearly as well as state-of-the-art methods despite a significant decrease in complexity.

4.1 Motivating the Generative Comparison Network

In this section, we motivate the Generative Comparison Network (GCN) by comparison to other few-shot learning algorithms.

To motivate the GCN, we consider the n-way, k-shot classification problem posed in Section 2.2, where given a labeled *support set*

$$\mathcal{S} := \{x_1^{(1)}, x_2^{(1)}, \ldots, x_k^{(1)}, x_1^{(2)}, \ldots x_k^{(n)}\} \subset \mathbb{R}^d$$

with n classes and k datapoints per class, we must identify which of the n classes a query point x_t belongs to.

For $x \in \mathbb{R}^d$ and $Q \subset \mathbb{R}^d$, we let $K(x, Q) : \mathbb{R} \times \mathcal{P}(\mathbb{R}) \to \mathbb{R}$ be a specified *similarity function* between an object and a set. Given such a similarity function, and letting $Q^{(i)} := \{x_1^{(i)}, x_2^{(i)}, \ldots, x_k^{(i)}\}$ be the subset of the support set with label i, a *metric-learning* style approach to the few-shot classification problem might produce the following distribution over the n possible labels for x_t:

$$p(x \in \text{class } i) = \frac{\exp(K(x, Q^{(i)}))}{\sum_{j=1}^{n} \exp(K(x, Q^{(j)}))} \tag{4.1}$$

One possible choice of $\mathcal{K}(x, Q)$ is the average Euclidean distance between the elements of Q and x. In practice, however, the Euclidean metric is not a good measure of true semantic difference between datapoints due to the complex, non-linear structure of many real-world datasets. Instead, in the fully-supervised setting, the Matching Networks algorithm ([38]) proposes first embedding x and the elements of Q into a lower-dimensional space using a shared neural network, and then computing $K(x, Q)$ as the average cosine similarity between the embedding of x and those corresponding to Q. The parameters of the neural network are learned via gradient descent on the log-loss of the prediction procedure on instances constructed from the training set.

Since its introduction, the original Matching Networks algorithm has been extended in a variety of ways, resulting in significant performance improvements on the original fully-supervised few-shot learning task. One of these ways is the Prototypical Networks algorithm [34], which makes two changes; firstly, the cosine similarity used to compare embeddings in Matching Networks is replaced with the Euclidean distance, which the authors argue is more geometrically sound. Second, when Q contains multiple elements, the authors propose using the similarity between the mean of the embeddings of elements in the set and the query embedding as $K(x, Q)$, as the mean better represents the characteristics shared within the set. In our development of the GCN, we will make use of both of these techniques.

Both Matching Networks and Prototypical Networks are unable to make use of unlabeled data during training. As previously observed, this limitation is particularly salient in many modern datasets. When operating in the semi-supervised setting, our proposed Generative Comparison Networks algorithm is a means of addressing this shortcoming by using the inference network of a variational autoencoder to generate embedding *distributions* of the query and elements of the support set, which are then used to compute $K^{GCN}(x, Q)$.

4.2 Model Description

In this section, we define the GCN and give the associated learning algorithm and loss function. Many of the design choices are motivated by the changes to the Latent Reasoning Network introduced in Chapter 3; we provide comparisons where relevant.

As described in Section 4.1, defining the GCN is solely a matter of specifying $K(x, Q)$. To do so, we begin with a Gaussian VAE with latent variable $z \in \mathbb{R}^l$, inference network $q_\phi(z|x)$, and recognition network $p_\psi(x|z)$. If $Q = \{x_1, x_2, \ldots, x_Q\}$, we define $q_i := q_\phi(z|x_i)$ for $1 \leq i \leq Q$, and $q_x = q_\phi(z|x)$. The collection of distributions $\{q_i\}$ and q_x will serve as embeddings of the support set and query, respectively.

To compare the embeddings, we will make use of a real-valued similarity function between distributions $D : \mathbb{P}(\mathbb{R}^k) \times \mathbb{P}(\mathbb{R}^k) \to \mathbb{R}$. The choice of similarity function can affect performance on the test-time task significantly in the point embedding setting, as illustrated by the discrepancy between Matching Networks and Prototypical Networks.

In our setting, embeddings also serve as posterior distributions for variational inference which is an added constraint complicating the choice of similarity function. For a simple example, note that algorithms dealing with point embeddings are free to scale all embeddings by a constant, whereas the scale for the GCN is fixed by the KL loss to the standard Gaussian prior. As a result, a change as simiple as scaling the GCN similarity function D by a constant could change the results non-trivially.

In the absence of a canonical means of comparing variational posterior distributions, in our experiments we evaluate each of a variety of choices for D listed below:

1. The **Bhattacharya Coefficient** (BC): $D(p_1, p_2) = \int_z \sqrt{p_1(z)p_2(z)}dz$
2. The **log-Bhattacharya Coefficient** (logBC): $D(p_1, p_2) = \log \int_z \sqrt{p_1(z)p_2(z)}dz$
3. The **mean-Euclidean Distance** (mED): $D(p_1, p_2) = ||\mathbb{E}_{p_1}[z] - \mathbb{E}_{p_2}[z]||\,|_2^2$
4. The **mean-Cosine Similarity** (mCS): $D(p_1, p_2) = \dfrac{\mathbb{E}_{p_1}[z] \cdot \mathbb{E}_{p_2}[z]}{||\mathbb{E}_{p_2}[z]||_2 ||\mathbb{E}_{p_2}[z]||_2}$
5. The **log-Latent Variable Bayes Factor** (LVBF): $D(p_1, p_2) = \log \int_z \dfrac{p_1(z)p_2(z)}{p(z)}dz$, where $p(z)$ is the PDF for $N(0, I_f)$.

When p_1, p_2 are distributed as $N(\mu_1, \Sigma_1)$ and $N(\mu_2, \Sigma_2)$ respectively, each of the choices of D described above has an analytic expression in terms of the μ_i and Σ_i, which are presented in Appendix A.

With D defined, we may now define $K^{GCN}(x, Q)$, which performs a single comparison between q_x and a *prototype* distribution constructed from the q_i (cf. Prototypical Networks), done as follows:

$$K^{GCN}(x, Q) = D(f(\{q_i\}_{1 \leq i \leq Q}), q_x) \tag{4.2}$$

For the GCN, a means of combining a set of distributions into single "prototype" distribution is necessary, represented by f. To motivate our choice of doing so, we introduce the $\mathcal{W}_2$ distance, a natural extension to distributions of the Euclidean distance for points.

For distributions $p_1, p_2 \in \mathbb{P}(\mathbb{R}^b)$, and $\pi \in \mathbb{P}(\mathbb{R}^b \times \mathbb{R}^b)$ with marginals p_1, p_2,

$$\mathcal{W}_2^2(p_1, p_2) = \inf_{\pi} \int ||x - y||_2^2 d\pi(x, y) \tag{4.3}$$

Mimicking the choice of the mean for Prototypical Networks, we choose the *Wasserstein barycenter* of the set as the prototype, which is the unique distribution minimizing the average Wasserstein distance to the elements. The Wasserstein barycenter has the following closed form when $q_i = \mathcal{N}(\mu_i, \Sigma_i)$ for diagonal Σ_i (as is the case in our setting), shown in [1]:

$$f(\{q_i\}_{1 \leq i \leq Q}) = \mathcal{N}(\overline{\mu}, \overline{\Sigma}) \tag{4.4}$$

where

$$\overline{\mu} = \frac{1}{Q} \sum_{i=1}^{Q} \mu_i \tag{4.5}$$

$$\overline{\Sigma} = \left(\frac{1}{Q} \sum_{i=1}^{Q} \sqrt{\Sigma_i} \right)^2 \tag{4.6}$$

We now give the loss function and the learning algorithm. The former will consist of a discriminative term $\mathcal{L}^{disc}$, designed to mimic the test-time few-shot classification task using the labeled training data, and an unsupervised term $\mathcal{L}^{ELBO}$ (Equation 2.1), making use of both the labeled and unlabeled datapoints to help reduce over-fitting, combined using a tuneable hyperparameter C.

For the discriminative term, we sample training instances $\mathcal{T}$ of the few-shot classification task from the labeled dataset $\mathcal{D}^L$ randomly using the following parameters: n' classes, with k' examples per class. Sampling proceeds by picking n' labels at random, which we call $1, 2, \dots n'$. For each of the selected labels i, we then select $k\prime$ datapoints $Q^{(i)} := \{x_1^{(i)}, x_2^{(i)}, \dots x_{k'}^{(i)}\}$ among those labeled i. Finally, an exemplar x_e is chosen among the as-of-yet unselected datapoints from a random class c among $\{1, 2, \dots n'\}$. $\mathcal{L}^{disc}$ is then defined as follows:

$$\mathcal{L}^{disc}(\phi) = \mathbb{E}_{\mathcal{T} \sim \mathcal{D}^L} \left[\log \left(\sum_{j=1}^{n} \exp(K(x_e, Q^{(j)})) \right) - K(x_e, Q^c) \right] \tag{4.7}$$

On its own, $\mathcal{L}^{disc}$ was first introduced as a means of training Matching Networks [38], and is widely used among other metric-learning approaches to few-shot learning such Prototypical Networks and Relation Nets. For the GCN, we add an unsupervised term:

$$\mathcal{L}^{unsup}(\phi, \psi) = \mathbb{E}_{x \sim \mathcal{D}^L \cup \mathcal{D}^U} [\mathcal{L}^{ELBO}(x, \phi, \psi)] \tag{4.8}$$

Our final loss function is then:

$$\mathcal{L}^{GCN}(\phi, \psi) = \frac{1}{C+1} \mathcal{L}^{unsup}(\phi, \psi) + \frac{C}{C+1} \mathcal{L}^{disc}(\phi) \tag{4.9}$$

We update parameters during training with noisy gradients estimated using stochastic gradient descent on $\mathcal{L}^{GCN}$.

We note that as $C \to \infty$, using the mean Euclidean Distance or mean Cosine Similarity as the similarity function recovers Prototypical Networks and Matching Networks respectively, as the generative loss disappears and the resulting loss pushes embeddings from the same class together and those from different classes apart. Accordingly, choosing increasing values of C starting at 0 allows us to interpolate between the results of using the VAE learned embeddings and the results for using the fully discriminative learning algorithms. As the amount of labeled data increases, we seek an algorithm closer and closer to the latter, providing intuitive justification for increasing C.

4.3 Experiments and Results

We evaluate the Generative Comparison Network algorithm along with the various proposed similarity scores on the semi-supervised few-shot learning task described in Section 3.6.3. As is standard, we give results in the 5-way setting for both $k = 1$ and $k = 5$.

To facilitate comparison, we use an identical convolutional VAE architecture and pretraining procedure to that used for the Omniglot experiments in Chapter 3. We set the number of training ways to $n' = 29$ and, due to the limited number of available images per class, set k' to 2. For each batch we sample 256 images for $\mathcal{L}^{unsup}$ and 32 training instances for $\mathcal{L}^{disc}$. We use the Adam optimizer with a learning rate of 0.0003, training for 8 epochs. The hyperparameter C is chosen via cross-validation separately for each similarity function. We find that setting C significantly higher or lower results in considerably worse classification performance.

4.3.1 Few-shot Learning Experiments

We show the results of the semi-supervised evaluation in Table 3.3. The GCN with the appropriate choice of similarity function considerably outperforms the fully-supervised baselines which don't make use of the unlabeled data, and is a slim margin below the improved LRN and the MetaGAN. We attribute the margin relative to the improved LRN to the permutation-equivariant layer of the reasoning network, which effectively allows the model to apply a transformation to the latent space prior to similarity calculation, increasing flexibility in the case of overly simplistic generative

Table 4.1: Accuracy on the 5-way Semi-supervised Omniglot task

Model	1-shot	5-shot
GCN (BC)	96.3	98.8
GCN (logBC)	96.3	98.7
GCN (mED)	93.8	96.5
GCN (mCS)	80.5	89.5
GCN (LVBF)	95.9	98.5
LRN (Chap. 3)	96.7	99.0
LRN [17]	92.4	97.2
MetaGAN [41]	97.1	-
Relation Net [35]	93.8	-
Prototypical Nets [34]	93.7	-

models. In contrast, the GCN does not use use any additional parameters to those already present in the pre-trained VAE, forcing comparisons to occur in the same space used for reconstruction. One possible means of closing this gap is through the use of more expressive latent variable models, which we discuss in 4.4.

In parallel to the discriminative setting, the choice of similarity function has a non-trivial impact on the performance of the GCN. In particular, the functions which depend solely on the distribution means (mED and mCS) perform relatively poorly, highlighting the importance of comparing the entire posterior distributions. A possible explanation for this discrepancy is that similarity functions solely based on the distribution means must weight distances between latent dimensions equally. When comparing general distributions, however, the ability to weight difference between the means of different dimensions based on the varibility of the distributions in those dimensions leads to better similarity judgements.

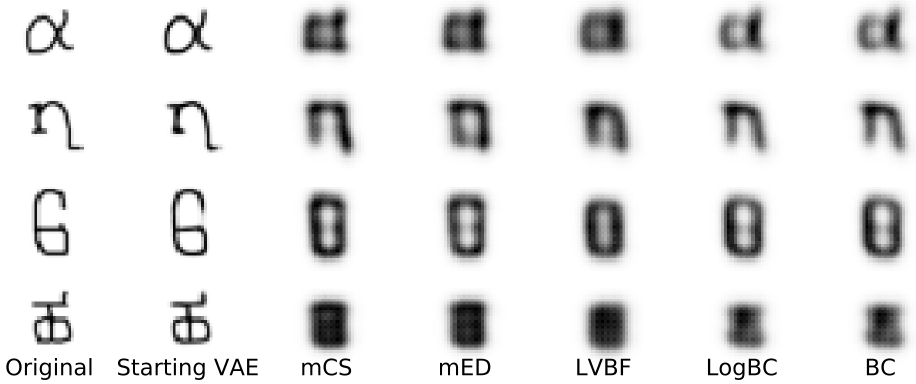

Figure 4-1: **Reconstructions using the Fine-tuned Generative Model :** Reconstructed characters for each choice of similarity function in addition to the pretrained VAE. As discriminative fine-tuning progresses, reconstruction quality suffers to a surprising extent, especially for similarity functions with low performance on the few-shot classification task. The top two characters are from the training set, while the bottom two are from the test set. To facilitate comparison, the parameter C controlling the weight of the discriminative loss relative to the generative loss is set to 50.0 for training of each of the models.

4.3.2 Qualitative Experiments

Here, we analyze the reconstructions produced by the fine-tuned VAE in more detail. While the primary goal of the fine-tuning procedure is to improve performance on the few-shot classification task, examining the generated characters provides a qualitative means of understanding modeling choices and tradeoffs. In particular, it seems reasonable that the fine-tuned model optimal for classification should also produce reasonable reconstructed characters. To understand why, note that the generative loss pushes latent posteriors of characters which are close visually together, while the discriminative loss pushes the embeddings of characters which are of the same class together. So long as characters within a class are visually similar, which is the case by construction, the tradeoff between these objectives should not be too large.

As observed in Figure 4-1, for all considered choices of the similarity function the quality of the reconstructed images is severely lacking when compared to those produced by the VAE parameters at the start of training. As explained previously, the likely culprit is the tension between the discriminative loss and generative loss caused by misspecification of the similarity function. If the similarity function is not suited to the geometry of the latent space, the latent posteriors may need to be warped more in order to decrease the discriminative loss, resulting in a steeper tradeoff. Additionally, the similarity functions that have the highest accuracy on the few-shot classification task (LogBC, BC) seem to have sharper reconstructions relative to the other. We hypothesize that this relationship is due to both the tradeoff and accuracy being driven by suitability of the similarity function, underscoring the importance of choosing a good similarity function.

4.4 Discussion

In the preceding sections, we introduced the *Generative Comparison Network*, an approach to classification problems which generalizes traditional fully-supervised metric-learning algorithms by means of a generative model. We showed the efficacy of the method on a semi-supervised and fully-supervised few-shot learning task. In the process, we displayed the trade-off between the quality of the generative model and the discriminative-ness of the embeddings.

Many design choices made for the Generative Comparison Network, such as the model structure and loss function, are in part motivated by their analogues in the Latent Reasoning Network described in Chapter 3. If we start with the Latent Reasoning Network, recovering the Generative Comparison Network algorithm is simply a matter of eschewing the hierarchical latent variable model and replacing the learned reasoning network with the parameter-less prototype computation heuristic described in 4.4. These changes permit us more flexibility in our choice of K for the GCN while also reducing the number of parameters necessary. As discussed for the Latent Reasoning Network, alternative parameterizations of the prior and posterior densities which produce a better or more disentangled generative model would likely result in improved performance on the few-shot classification task and are certainly worth exploring.

A variety of other interesting avenues of exploration exist. The metric-learning paradigm used for the GCN and other few-shot learning algorithms, in which classification is performed via distance between learned embeddings, may also be used for standard classification problems in which the labels in the training set are also present in the test set. While such approaches are unlikely to outperform label prediction-based methodologies in settings where each label is represented by large quantities of data, this might not be the case in extreme classification, in which an extremely large number of labels with few datapoints each are available during training.

Additionally, the choices of similarity function and prototype for our method are largely based on heuristics drawn from the fully-supervised setting. Testing alternatives, especially if justified theoretically, could be beneficial. Another option is moving towards a learned similarity function, which results in significant gains in the fully-supervised setting ([35]).

Chapter 5

Disentanglement via Decorrelation

In this chapter, we present a simple, computationally efficient algorithm for disentangling a Gaussian VAE generative model after training. If we permit full, rather than diagonal, covariance matrices for individual posteriors $q_\phi(z|x)$, we prove that a variant of our algorithm results in a strictly improved ELBO. The lack of a tradeoff between generative model quality, as measured by the ELBO, and a form of disentanglement (albeit weak) stands in contrast to previous work.

We evaluate the proposed algorithm on a variety of benchmarks, and compare to a collection of existing VAE variants designed to produce disentangled representations. To begin, we formalize disentanglement and discuss prior approaches.

5.1 Motivating Decorrelation

We consider the problem of learning low-dimensional representations which are human-interpretable and useful for downstream tasks via variational autoencoders. Representations in which certain dimensions of the representations recover known factors of variation within the dataset are a step towards achieving these properties. For example, a handwriting dataset might count digit class, rotation, and stroke thickness among its factors of variation. A natural question to ask is how such interpretable (or "disentangled") representations may be learned in an unsupervised manner.

Based on the intuition that moving along an axis corresponding to one factor of variation should not influence another, one condition thought to be necessary for disentanglement is statistical independence of the latent dimensions. To formalize this notion, we define the *aggregate posterior* $q^{agg}(z)$:

$$q^{agg}(z) = \sum_{i=1}^{N} \frac{1}{N} q_{\phi}(z|x_i) \tag{5.1}$$

The aggregate posterior is the marginal distribution of the latent variables z given the datapoints and fixed encoder, and so statistical independence of the latent dimensions is synonymous with independence of the individual dimensions within the aggregate posterior. To emphasize this property in trained VAEs, recent approaches add a additional regularization term with an associated weighting hyperparameter to the standard VAE loss function (as defined in Equation 2.1).

Among these, the Beta-VAE [11] increases the weight of the KL term within the normal VAE objective, under the reasoning that forcing individual posteriors to be closer to the standard Gaussian prior will result in independence. As a refinement to this approach, the FactorVAE [13] and TC-VAE [2] penalize the *total correlation*,

$$TC(q^{agg}) = KL\left(q^{agg}(z)||\prod_{i} q^{agg}(z_i)\right)$$

a generalization of mutual information to more than two variables of the aggregate posterior. The total correlation is reduced as the dimensions of the aggregate posterior become more independent. As the aggregate posterior is a mixture of Gaussians with number of components equal to the number of datapoints, this is computationally intractable to compute repeatedly. To address this issue, the FactorVAE and TC-VAE use noisy estimators of the total correlation, based on an auxiliary classifier and a biased minibatch estimation procedure, respectively.

Instead of taking a mutual-information approach, Kumar et. al [19] suggest relaxing the independence condition on the aggregate posterior to simply matching the first two moments of the prior. To do so, the DIP-VAE-I and DIP-VAE-II penalize the squared Frobenius norm of the difference between the empirical covariance matrix of the sampled minibatch and that of the prior.

Our model will achieve the DIP-VAE goal of matching the first and second moments of the aggregate posterior to those of the prior via a slightly different approach. To begin, we show that these moments may be easily computed.

5.2 Computing the Aggregate Covariance

When the outputs of the encoder are Gaussian distributions, many important statistics of the resulting aggregate posterior

may be expressed in a simple closed form. In particular, taking $\mu = \frac{1}{N}\sum_{i=1}^{N} \mu_{x_i}$ the following holds:

$$\Sigma_{agg} := \mathrm{Cov}(q^{agg}) = \frac{1}{N}\sum_{i=1}^{N} \left[(\mu_{x_i} - \mu)(\mu_{x_i} - \mu)^T + \Sigma_{x_i}\right] \tag{5.2}$$

When training Gaussian VAEs, we observe aggregrate covariances which are close to, but nonetheless depart nontrivially from the identity matrix.

5.3 Decorrelating the Latent Space

In this section, we will describe our method, which consists of two steps: 1) computing the second-order moments of the aggregate posterior, and 2) producing a modified model with an identity covariance matrix. We prove that the resulting model, while achieving an identity covariance, also results in an improved ELBO.

We start with a dataset $\mathcal{D} := \{x_1, x_2, ..., x_N\} \subset \mathbb{R}^d$ and latent space $\mathbb{R}^l$. Our analysis will be concerned with the commonly-used Gaussian variational autoencoder, as described in Section 2.1, with inference network $q_\phi(z|x) = N(\mu_x, \Sigma_x)$ and recognition network $p_\psi(x|z) = f(z)$. For simplicity, we omit the subscripts.

In practice, both networks are commonly parameterized by deep neural networks, which affords flexibility in modeling complex, real-world datasets with nonlinear structure. Additionally, the Σ_x are typically taken to be diagonal matrices. Our algorithm will operate independent of these assumptions.

To begin, we calculate Σ_{agg} for the original model, which requires a single forward pass through the encoder for each datapoint. For the second step, taking $A = \sqrt{\Sigma_{agg}}^{-1}$, we define the modified VAE, which we label the *CorrVAE*, as follows:

$$\begin{aligned} q'(z|x) &= N(A\mu_x - A\mu, A\Sigma_x A^T) \\ p'(x|z) &= f(A^{-1}(z + \mu)) \end{aligned} \tag{5.3}$$

We formalize our claims above in this proposition:

Proposition 2. *Under the CorrVAE as defined above, the following hold:*

1. ***(Identity Aggregate Covariance)***

   The aggregate covariance matrix of the modified VAE is I_l

2. ***(Identical Reconstruction Loss)***

   For all x, $\mathbb{E}_{z\sim q'(z|x)}[\log p'(x|z)] = \mathbb{E}_{z\sim q(z|x)}[\log p(x|z)]$

3. ***(Improved Training KL Loss)***

   $\frac{1}{N}\sum_{i=1}^{N} KL(q'(z|x_i)||p(z)) \leq \frac{1}{N}\sum_{i=1}^{N} KL(q(z|x_i)||p(z))$

Proof. For 1, the new aggregate covariance matrix, denoted by ${\Sigma_{agg}}'$, depends solely on q'. By 5.2,

$$
\begin{aligned}
{\Sigma_{agg}}' &= \frac{1}{N}\sum_{i=1}^{N}\left[(A\mu_{x_i} - A\mu)(A\mu_{x_i} - A\mu)^T + A\Sigma_{x_i}A^T\right] \\
&= A\left(\frac{1}{N}\sum_{i=1}^{N}\left[(\mu_{x_i} - \mu)(\mu_{x_i} - \mu)^T + \Sigma_{x_i}\right]\right)A^T \\
&= A\Sigma_{agg}A^T \\
&= I_k
\end{aligned}
$$

where the last equality follows from A being symmetric, as the inverse square root of a symmetric matrix.

For 2, note that if $z \sim q(z|x)$, then $Az - A\mu \sim q'(z|x)$. Accordingly,

$$
\begin{aligned}
\mathbb{E}_{z\sim q'(z|x)}[\log p'(x|z)] &= \mathbb{E}_{z\sim q(z'|x)}[\log f(A^{-1}z + \mu)] \\
&= \mathbb{E}_{z\sim q(z|x)}[\log f(A^{-1}(Az - A\mu) + \mu)] \\
&= \mathbb{E}_{z\sim q(z|x)}[\log p(x|z)]
\end{aligned}
$$

as desired.

For 3, consider the family of encoders

$$
\mathcal{Q} := \{r(\cdot|x) | r(\cdot|x) \sim N(S(\mu_x - v), S\Sigma_x S^T), S \in \mathbb{R}^{l\times l}, v \in \mathbb{R}^l\} \qquad (5.4)
$$

Clearly $q, q' \in \mathcal{Q}$, taking $S = I_k, v = 0$ and $S = A, v = \mu$, respectively.

Now, we show that q' is in fact the minimizer of the KL loss over this family. Using the closed-form expression for the KL-divergence between two multivariate Gaussians, and taking $U = S^T S$

$$\begin{aligned}
&\operatorname*{arg\,min}_{S \in \mathbb{R}^{k\times k}, v \in \mathbb{R}^l} \frac{1}{N}\sum_{i=1}^{N} KL(r(z|x_i; S, v)||p(z)) \\
&= \operatorname*{arg\,min}_{S \in \mathbb{R}^{k\times k}, v \in \mathbb{R}^l} \frac{1}{N}\sum_{i=1}^{N} KL(N(S\mu_{x_i} - Sv, S\Sigma_{x_i}S^T)||N(0, I_k)) \\
&= \operatorname*{arg\,min}_{S \in \mathbb{R}^{k\times k}, v \in \mathbb{R}^l} \frac{1}{N}\sum_{i=1}^{N} -\log\det(S\Sigma_{x_i}S^T) + tr(S\Sigma_{x_i}S^T) + (\mu_{x_i} - v)^T S^T S(\mu_{x_i} - v) \\
&= \operatorname*{arg\,min}_{S \in \mathbb{R}^{k\times k}, v \in \mathbb{R}^l} \frac{1}{N}\sum_{i=1}^{N} -\log\det(S^T S) + tr(S^T S\Sigma_{x_i}) + tr(S^T S(\mu_{x_i} - v)(\mu_{x_i} - v)^T) \\
&= \operatorname*{arg\,min}_{U \in \mathbb{R}^{k\times k}, v \in \mathbb{R}^l} \frac{1}{N}\sum_{i=1}^{N} -\log\det(U) + tr(U\Sigma_{x_i}) + tr(U(\mu_{x_i} - v)(\mu_{x_i} - v)^T)
\end{aligned}$$

where the last equality follows from the multiplicativity of the determinant and the cyclic property of the trace. The resulting optimization problem is convex and smooth in each argument, as each individual term is convex and smooth in each argument. Additionally, note that $U = S^T S \succ 0$, as otherwise $-\log\det(U)$ is infinite. We may therefore characterize the minimizers setting the derivatives with respect to U and v to 0 and solving. Proceeding,

$$\begin{aligned}
0 &= \frac{d}{dv}\frac{1}{N}\sum_{i=1}^{N} -\log\det(U) + tr(U\Sigma_{x_i}) + (\mu_{x_i} - v)^T U(\mu_{x_i} - v) \\
0 &= \frac{1}{N}\sum_{i=1}^{N} 2v^T S - 2\mu_{x_i}^T S \\
0 &= v^T - \frac{1}{N}\sum_{i=1}^{N} -\mu_{x_i}^T \\
v &= \mu
\end{aligned}$$

where the third equality is obtained by using the invertibility of U.

Taking $v = \mu$,

$$
\begin{aligned}
0 &= \frac{d}{dU}\frac{1}{N}\sum_{i=1}^{N} - \log\det(U) + tr(U\Sigma_{x_i}) + tr(U(\mu_{x_i} - \mu)(\mu_{x_i} - \mu)^T) \\
0 &= -U^{-1} + \frac{1}{N}\sum_{i=1}^{N}\left[(\mu_{x_i} - \mu)(\mu_{x_i} - \mu)^T + \Sigma_{x_i}\right] \\
U &= \Sigma_{agg}^{-1}
\end{aligned}
$$

Accordingly, the training KL loss is minimized precisely when $S^T S = \Sigma_{agg}^{-1}$ and $v = \mu$. Taking $S = A$ satisfies this condition, as desired. □

As the training ELBO decomposes as the sum of the reconstruction loss and the KL loss, the proposition above gives the following informally stated corollary:

Corollary 2. *The modified VAE achieves a better training ELBO, while also achieving decorrelation.*

Additionally, the sole property of A necessary to prove Proposition 1 was that $A^T A = \Sigma_{agg}^{-1}$. In particular, for an arbitrary orthonormal matrix $R \in \mathbb{R}^{l \times l}$, $(RA)^T(RA) = \Sigma_{agg}^{-1}$. For any such R, the following stronger statement therefore holds:

Corollary 3. *Proposition 2 holds for the following VAE:*

$$
\begin{aligned}
q'(z|x) &= N(RA\mu_x - RA\mu, RA\Sigma_x(RA)^T) \\
p'(x|z) &= f((RA)^{-1}(z + \mu))
\end{aligned}
\tag{5.5}
$$

We discuss the implications of this corollary in Section 5.5.

5.4 Experiments and Results

We now evaluate the CorrVAE quantitatively on a variety of metrics of interest and compare to state-of-the-art methods for disentanglement. We are concerned with our model's relative and absolute performance in three broad categories: quality of the trained generative model, closeness of the aggregate posterior to the prior (*unsupervised* disentanglement), and recovery of the ground truth factors of variation (*supervised* disentanglement).

To facilitate reproducibility and comparison with previous work, we implement our method within the `disentanglement_lib` Python library [21], which provides a collection of pre-trained models with various hyperparameters along with code to measure many quantities of interest given a trained model. A single simple convolutional encoder/decoder architecture with 10 latent variables is used across all models (including the CorrVAE), and identical training hyperparameters are used to whatever extent possible to negate the effect of different architectural choices. All values for methods besides the CorrVAE are computed using the available pre-trained models.

In choosing and labeling regularization strength, we adopt the conventions of [21]; namely, a range of 6 hyperparameters based on the recommendations of the model's original paper are evaluated. To simplify visualization, these hyperparameters are identified in plots by the sequence $\{0, 0.2, ..., 1\}$, however these values are not directly comparable across models.

For our experiments, we use the dSprites dataset [25], explicitly created for evaluating disentanglement methods. Each datapoint is a 64×64 black-white image created using a particular setting of five known factors of variation: shape, x/y position, rotation, and scale (see Figure 5-1). An image corresponding to every possible combination of these factors is present in the dataset, for which there does not exist a separate training and test set.

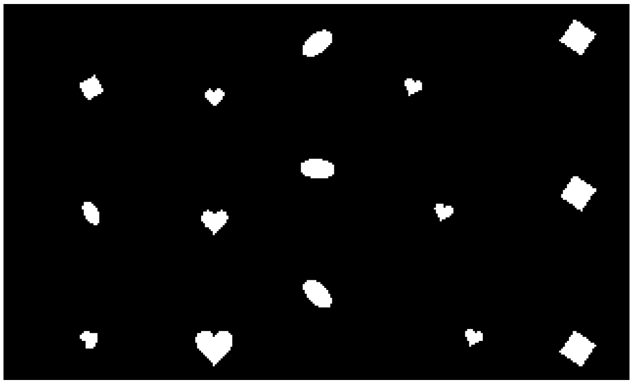

Figure 5-1: **dSprites Visualization:** Image samples from the dSprites dataset are shown, with one factor of variation varied in each column. From left to right: shape, scale, rotation, x coordinate, y coordinate.

5.4.1 Generative Model Quality

Our experiments evaluating the evidence lower bound, reconstruction loss, and KL loss for the CorrVAE agree with Proposition 2, and are shown in figure 5-2. In particular, compared to the VAE, the CorrVAE achieves an identical reconstruction loss (ignoring sampling error) and marginally lower KL loss, resulting in a lower training ELBO. For all other disentanglement methods, increasing the weight of the disentanglement term (i.e. regularization strength) results in a substantial deterioration in reconstruction quality and the ELBO score.

5.4.2 Unsupervised Disentanglement Metrics

Now, we turn to evaluating the learned representations themselves. Take $r(x) \in \mathbb{R}^{\lessdot}$ to be the representation of a particular point, defined as either a sample from or the mean of the corresponding distribution outputted by the encoder. The independence of the dimensions in the representations across the data is commonly referred to as a necessary (or even sufficient) condition for disentanglement. Following [21], we

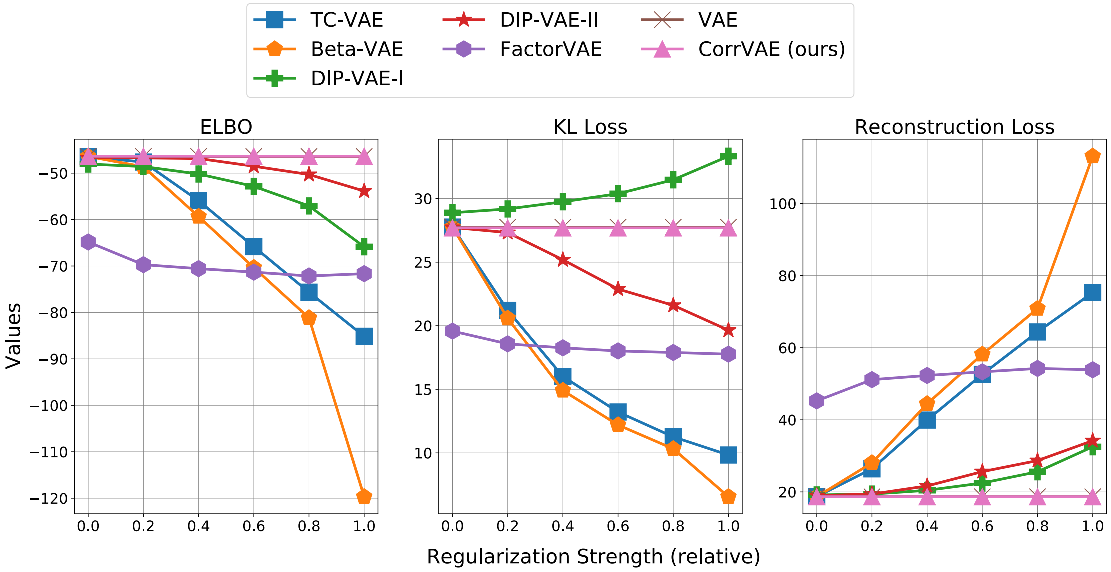

Figure 5-2: **Evaluating the Trained Generative Model:** Unsupervised metrics evaluated on the training set, averaged across 50 runs (For reconstruction loss and KL loss, higher is worse). As regularization strength increases, all previous models see a significant reduction in quality as measured by the ELBO. Both the increasing reconstruction loss and decreasing KL loss from left to right is a result of predicting individual posteriors which grow closer to the prior.

evaluate the former via the following two quantities:

The **Gaussian Total Correlation**: We sample 10,000 datapoints and take the mean $\mu_{r(x)}$ and empirical covariance $\Sigma_{r(x)}$ of their representations. The Gaussian Total Correlation is $KL(N(\mu_{r(x)}, \Sigma_{r(x)})||\prod_i N(\mu_{r(x)_i}, \Sigma_{r(x)_{ii}}))$. Note that this is a Gaussian approximation to the aggregate posterior. The resulting value is low if the representations are uncorrelated, which is a necessary (albeit insufficient) condition for independence.

The **Mutual Information Score**: To compute the Mutual Information Score, we begin by sampling 10,000 datapoints. For a particular pair of dimensions, each dimension is paritioned into 20 bins, and the quantity of datapoints falling into each pair of bins is used to compute the discrete mutual information. This quantity is averaged across each possible pair of dimensions to produce the Mutual Information Score, which serves as a heuristic measure of dependence able to take into account nonlinear relationships between dimensions.

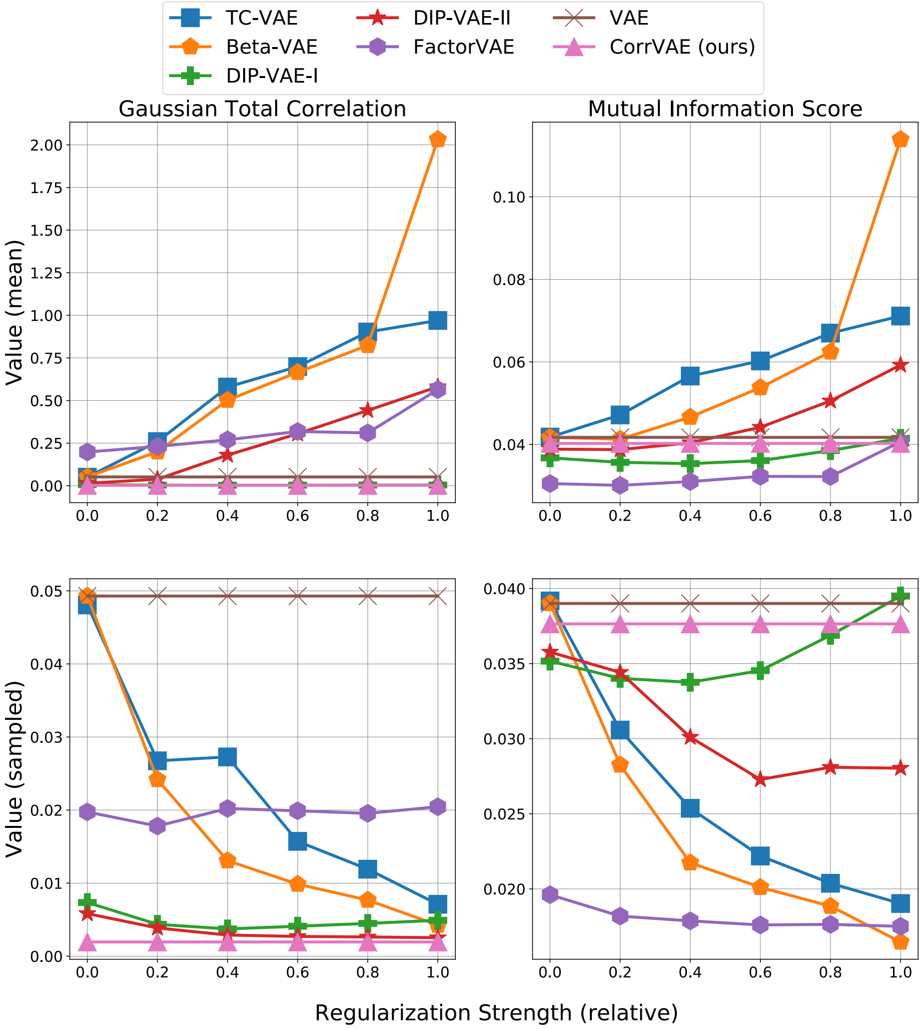

Figure 5-3: **Evaluating Disentanglement (Unsupervised):** Unsupervised disentanglement metrics evaluated on the training set, averaged across 50 runs (higher is worse). For all previous methods besides the DIP-VAE-I, the discrepancy in trends between the mean and sampled representations is a result of the regularization term being estimated using *samples* from the individual posterior, rather than the mean. As a result, samples from the aggregate posterior increasingly resemble samples from the prior distribution, whereas the means of the individual posteriors don't necessarily.

Our model performs well on the Gaussian total correlation for both choices of representation, as we enforce the independence condition on the first and second moments. We observe that the mutual information score for the CorrVAE is worse for the sampled representations than most other models (while still displaying an improvement on the vanilla VAE). This is partially a result of the other method's ability to penalize higher order moments of the aggregate posterior, for which the sampled representation is a rough approximation. Importantly, the mutual information score for all other models worsens significantly when the representation is taken to be the mean, as is common in practice. This suggests that other models are partially achieving this low score for sampled representations via large individual posterior variances, which in turn is in agreement with their poor reconstruction losses.

5.4.3 Supervised Disentanglement Metrics

Using `disentanglement_lib`, we evaluate CorrVAE on a variety of supervised disentanglement metrics put forward in the literature, with results shown in figure 5-4. All of these metrics are different means to measure disentanglement using ground truth factors of variation. We give a brief description of each below: for additional detail on their computation, we point the reader towards their respective papers and [21]. All classifiers use a training set of size 10000 and are evaluated on a test set of size 5000.

1. **Mutual Information Gap (MIG)** [2]: The MIG is the difference between in mutual information between the first and second most informative latent dimensions normalized by the entropy of the factor of variation, averaged over all factors of variation. Models in which a single latent dimension correspond to single factors of variation have a higher MIG.

2. **SAP Score** [19]: For each factor, a classifier is trained for each latent dimension to predict the factor value. The SAP score is the difference in out-of-sample accuracy between the classifiers corresponding to the two most predictive di-

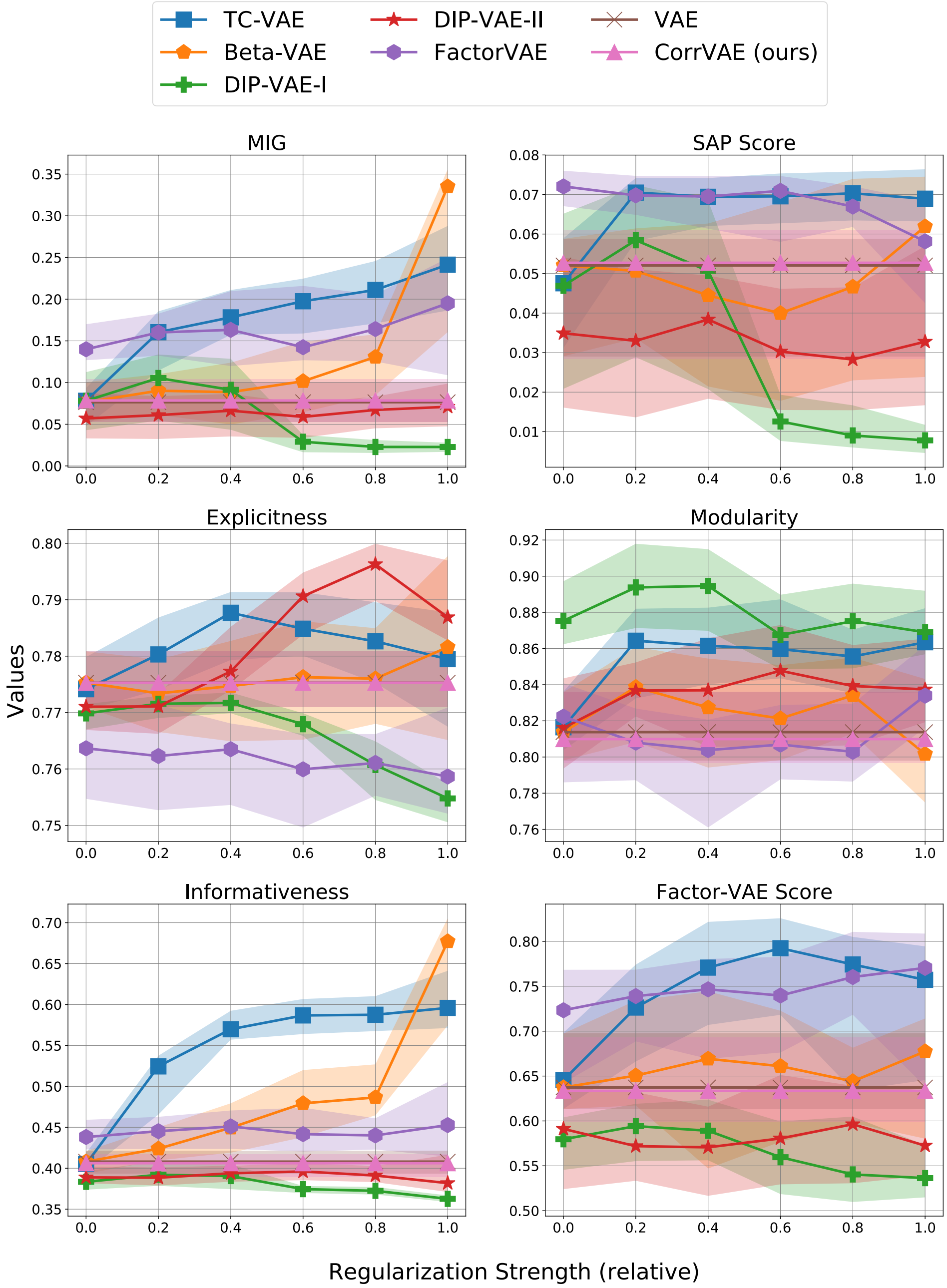

Figure 5-4: **Evaluating Disentanglement (Supervised):** Supervised disentanglement metrics evaluated for different models. Median values are plotted with the 25%/75% quantiles shaded. As observed in [21], there is no clear optimal objective function across all metrics, with significant variance attributable to the choices of random seed and regularization hyperparameter.

mensions, averaged across factors. Models in which a single latent dimension correspond to single factors of variation have a higher SAP score.

3. **Explicitness** [32]: The out-of-sample accuracy of a logistic regression classifier trained to predict a particular factor of variation given the latent vector, averaged over factors of variation. Models in which the latent representations are predictive of the factors of variation are more explicit.

4. **Modularity** [32]: The squared difference in mutual information between the first and second most informative latent dimensions normalized by the entropy of the factor of variation, averaged over all factors of variation. Models in which a single latent dimension correspond to single factors of variation are more modular.

5. **Informativeness** [4]: The accuracy of a random forest classifier trained to predict a particular factor based on the latent vector, averaged across factors. Models in which the latent representations are predictive of the factors of variation are more explicit.

6. **Factor-VAE Score** [13]: To begin, we sample a set of datapoints with a fixed value of a single unknown factor of variation. The Factor-VAE score is the accuracy in predicting the fixed factor of variation of a classifier which, based on the index of the latent dimension of least variance of such a set, outputs the factor of variation. The majority vote classifier is trained using sets sampled from the training set.

We observe that our method performs similarly to the normal VAE for all metrics. This is largely a result of most methods involving the performance of a classifier (typically linear or tree-based) trained on the representations, which is not affected by linear transformations of the representation.

The CorrVAE outperforms the DIP-VAE variants on nearly every metric besides modularity, suggesting the promise of our approach to decorrelation. While the CorrVAE is subpar relative to the mutual-information-based methods on these metrics, the comparatively good ability of the CorrVAE to produce an independent aggregate posterior suggests that the results of the other methods are not *because* they penalize dependence between dimensions. Instead, their performance may be due to an unintended result of the inductive biases. Regardless, this phenomenon merits further investigation, as it goes against common intuition regarding disentanglement.

5.5 Discussion

Our results suggest that the CorrVAE is able to learn representations which compare favorably to other methods in inducing independence between latent dimensions, while provably avoiding any reconstruction-loss tradeoff. Importantly, training the CorrVAE consists of a single post-processing step after the training of a vanilla VAE, avoiding any hyperparameter tuning. While performance on supervised metrics is lacking, this suggests that the independence condition on the aggregate posterior, while possibly necessary, is far from sufficient for disentanglement, which is particularly interesting.

One point of difference between the CorrVAE and other methods is the use of a non-diagonal posterior covariance. We observe that during training, diagonal posterior covariances are still used, so no additional computational cost is incurred. Additionally, nearly all metrics are computed using the mean of the individual posterior as the representation, which is not dependent on the parameterization of the covariance.

There are multiple potential future directions of interest. While computing the inverse covariance matrix analytically is typically not possible for non-Gaussian choices of individual posterior, accurate approximations may still be obtained via Monte Carlo estimation. Once such an empirical estimate is obtained, an identical procedure to CorrVAE may be applied to transform samples from the original individual posterior to achieve decorrelation.

An additional direction stems from the observation that the choice of decorrelation transformation is invariant to rotations of the latent space, as described in Corollary 5.5. A natural extension is to choose a particular rotations to optimize a particular objective. A variety of choices exist for this objective. If conditional independence of the latent variables is desired, we might choose a rotation which minimizes the off-diagonal entries of the individual posteriors. If a small set of labels of ground-truth factors of variation are available, as in the *weakly supervised* disentanglement setting [22], a rotation might be chosen so as to maximize alignment between the learned latent variables and the labeled factors of variation, by any of the supervised metrics mentioned in section 5.4.

Lastly, a more ambitious objective is for disentangled models to not only be useful for downstream tasks, but also to mimic the human process of reasoning about objects. In order to achieve this, however, we must first understand the properties of the set of optimal model parameters under our objective given arbitrary amounts of data. This process is greatly simplified when the space consists only of a single parameter set, in which case the underlying model is referred to as *identifiable.* While sufficient conditions for identifiability are known for a variety of linear factor analysis models [33], relatively little is known for their non-linear extensions, such as deep generative models. An understanding of how to modify existing objectives or parameterizations, such as those proposed by the CorrVAE and other disentanglement methods, to lead to identifiability in the non-linear factor analysis is needed.

Chapter 6

Discussion

In this thesis, we proposed a variety of methods which fine-tune a trained generative model to produce disentangled latent spaces or boost few-shot classification performance. Through empirical, and in the case of the former, theoretical analyses we demonstrated the promise and tradeoffs of our approaches against existing algorithms. In this section, we briefly summarize some promising research directions.

For all of our methods for few-shot learning, we fine-tune a Gaussian variational autoencoder, a choice made widely throughout the literature due to its simplicity. As discussed in Section 3.7, many alternatives to the Gaussian VAE have been proposed which result in a generative model with higher-fidelity samples and tighter lower bounds on the marginal likelihood. For few-shot classification, this may produce a more refined notion of similarity, allowing the unsupervised portion of our proposed loss function to more effectively leverage unlabeled examples. There is, however, a cost to abandoning Gaussianity. The likelihood functions used as similarity measures for our methods, such as the Latent Variable Bayes Factor, lack a closed form for many more complex distributions. To evaluate such a function therefore would likely require Monte Carlo estimation, which can have large variance when dealing with high-dimensional distributions.

Most common methods for disentanglement, including ours, have similarly built on the Gaussian VAE framework. The authors of [24] suggest the rotation-invariance of the standard Gaussian prior may render it unsuitable for disentanglement, as the training objective is rotation-invariant while a fully disentangled representation is not. To address this issue, the Student t-Distribution is proposed as an alternative. More broadly, an understanding of how the desired properties of a disentangled representation may be encoded by a choice of prior remains incomplete. Changes to the parameterization of the individual posterior could likewise produce positive results.

Part of the difficulty inherent in disentangled representation learning is the lack of a formal definition of disentanglement. As a result, identifying suitable inductive biases at training-time is difficult, with widespread acceptance of disentanglement as synonymous with a factorized aggregate posterior. As the evaluation of the CorrVAE algorithm in Section 5.4 shows, this unsupervised objective does not necessarily correspond to strong performance on supervised disentanglement metrics. That said, the TC-VAE is nonetheless able to perform better than a standard VAE in recovering particular factors of variation, despite consisting of an objective whose sole articulated purpose is to make the dimensions of the aggregate posterior independent. This suggests that there is more subtle reason for its success, which is worth studying further.

On a more practical level, despite being a primary motivation for their development the capability of disentangled representation learning methods to help humans work with real-world datasets has largely remained unexplored. In particular, one setting where such methods could be valuable is in modeling the graph-based structures of molecules. There has been significant recent interest in VAE-based unsupervised-learning approaches to this problem, with the learned representations useful in predicting chemical properties and generating new molecules with desirable properties [8]. Combining these models with the CorrVAE procedure could result in a representation space with an increased correspondence between individual properties and latent dimensions, facilitating the process of generating molecules with particular attributes and simplifying human understanding of the relationships between molecules.

Another use for disentanglement methods is in settings where labeled data is scarce. After using such a method, we might expect class identity to be captured in only a restricted subset of the latent variables, rather than across all dimensions. As a result, the prediction problem is made sparse, potentially improving the sample complexity of learning.

Appendix A

Closed Form Similarity Functions

Here, we give closed forms for each of the choices of D described in Section 4.2. We assume the Gaussian distributions p_1 and p_2 have means $\mu^{(1)} = \langle \mu_1^{(1)}, \mu_2^{(1)}, \ldots, \mu_k^{(1)} \rangle$ and $\mu^{(2)} = \langle \mu_1^{(2)}, \mu_2^{(2)}, \ldots, \mu_k^{(2)} \rangle$ respectively, and diagonal covariance matrices $\Sigma^{(1)} = \mathrm{diag}(\langle (\sigma_1^{(1)})^2, (\sigma_2^{(1)})^2, \ldots, (\sigma_k^{(1)})^2 \rangle)$ and $\Sigma^{(2)} = \mathrm{diag}(\langle (\sigma_1^{(2)})^2, (\sigma_2^{(2)})^2, \ldots, (\sigma_k^{(2)})^2 \rangle)$ respectively.

1. **Bhattacharya Coefficient** (BC): Letting $\mu' = (\Sigma^{(1)})^{-1}\mu^{(1)} + (\Sigma^{(2)})^{-1}\mu^{(2)}$ and $\Sigma' = \left((\Sigma^{(1)})^{-1} + (\Sigma^{(2)})^{-1}\right)^{-1}$,

$$\int_z \sqrt{p_1(z)p_2(z)}dz = \sqrt{\frac{2^k|\Sigma'|}{\sqrt{|\Sigma^{(1)}||\Sigma^{(2)}|}}} \\ \times \exp\left(-\frac{1}{4}\left((\mu^{(1)})^T(\Sigma^{(1)})^{-1}(\mu^{(1)}) + (\mu^{(2)})^T(\Sigma^{(2)})^{-1}(\mu^{(2)}) - (\mu')^T(\Sigma')^{-1}(\mu')\right)\right)$$

2. **log-Bhattacharya Coefficient** (logBC):

$$\log \int_z \sqrt{p_1(z)p_2(z)}dz = \frac{k\log 2}{2} + \frac{\log|\Sigma'|}{2} \\ -\frac{1}{4}\left(\log|\Sigma^{(1)}| + \log|\Sigma^{(2)}| + (\mu^{(1)})^T(\Sigma^{(1)})^{-1}(\mu^{(1)}) + (\mu^{(2)})^T(\Sigma^{(2)})^{-1}(\mu^{(2)}) - (\mu')^T(\Sigma')^{-1}(\mu')\right)$$

3. **mean-Euclidean Distance** (mED):

$$||\mathbb{E}_{p_1}[z] - \mathbb{E}_{p_2}[z]||_2^2 = ||\mu_1 - \mu_2||_2^2$$

4. **mean-Cosine Similarity** (mCS):

$$\frac{\mathbb{E}_{p_1}[z] \cdot \mathbb{E}_{p_2}[z]}{||\mathbb{E}_{p_2}[z]||_2 ||\mathbb{E}_{p_2}[z]||_2} = \frac{\mu_1 \cdot \mu_2}{||\mu_1||_2 ||\mu_2||_2}$$

5. The **log-Latent Variable Bayes Factor** (LVBF): Taking $A_i = (\sigma_i^{(1)})^2 + (\sigma_i^{(2)})^2 - (\sigma_i^{(1)})^2(\sigma_i^{(2)})^2$ and $p(z)$ to be the PDF of a standard normal random variable,

$$\log \int_z \frac{p_1(z)p_2(z)}{p(z)} dz = \sum_{i=1}^{k} \left(\frac{-(\mu_i^{(1)} - \mu_i^{(2)})^2 + (\mu_i^{(1)})^2(\sigma_i^{(2)})^2 + (\mu_i^{(2)})^2(\sigma_i^{(1)})^2}{2A_i} - \log \sqrt{A_i} \right)$$

For a full derivation of the closed form of the Bhattacharya Coefficient, we refer the reader to [12]. The expressions for the mean-Euclidean Distance and mean-Cosine Similarity follow directly from their definitions, and the derivation for the LVBF may be found in Section 3.3.

Bibliography

[1] Martial Agueh and Guillaume Carlier. Barycenters in the wasserstein space. *SIAM Journal on Mathematical Analysis*, 43(2):904–924, 2011.

[2] Tian Qi Chen, Xuechen Li, Roger B Grosse, and David K Duvenaud. Isolating sources of disentanglement in variational autoencoders. In *Advances in Neural Information Processing Systems*, pages 2610–2620, 2018.

[3] Emilien Dupont. Learning disentangled joint continuous and discrete representations. In *Advances in Neural Information Processing Systems*, pages 710–720, 2018.

[4] Cian Eastwood and Christopher KI Williams. A framework for the quantitative evaluation of disentangled representations. In *ICLR*, 2018.

[5] Harrison Edwards and Amos Storkey. Towards a neural statistician. In *ICLR*, 2017.

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. *ICML*, 2017.

[7] Zoubin Ghahramani and Katherine A Heller. Bayesian sets. In *NIPS*, 2005.

[8] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[10] Luke Hewitt, Maxwell Nye, Andreea Gane, Tommi Jaakkola, and Joshua Tenenbaum. The variational homoencoder: Learning to learn high capacity generative models from few examples. *arXiv preprint arXiv:1602.08919*, 2018.

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

[12] Tony Jebara, Risi Kondor, and Andrew Howard. Probability product kernels. *Journal of Machine Learning Research*, 5(Jul):819–844, 2004.

[13] Hyunjik Kim and Andriy Mnih. Disentangling by factorising. *arXiv preprint arXiv:1802.05983*, 2018.

[14] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. In *ICLR*, 2014.

[15] Durk P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581–3589, 2014.

[16] Gregory Koch. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning Workshop*, 2015.

[17] Rahul Krishnan, Arjun Khandelwal, Rajesh Ranganath, and David Sontag. Max-margin learning with the bayes factor. In *UAI*, 2018.

[18] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[19] Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations. *arXiv preprint arXiv:1711.00848*, 2017.

[20] Brenden M Lake, Ruslan R Salakhutdinov, and Josh Tenenbaum. One-shot learning by inverting a compositional causal process. In *NIPS*, 2013.

[21] Francesco Locatello, Stefan Bauer, Mario Lucic, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. *arXiv preprint arXiv:1811.12359*, 2018.

[22] Francesco Locatello, Michael Tschannen, Stefan Bauer, Gunnar Rätsch, Bernhard Schölkopf, and Olivier Bachem. Disentangling factors of variation using few labels. *CoRR*, abs/1905.01258, 2019.

[23] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

[24] Emile Mathieu, Tom Rainforth, Siddharth Narayanaswamy, and Yee Whye Teh. Disentangling disentanglement. *arXiv preprint arXiv:1812.02833*, 2018.

[25] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset. https://github.com/deepmind/dsprites-dataset/, 2017.

[26] Tsendsuren Munkhdalai and Hong Yu. Meta networks. *ICML*, 2017.

[27] Jason W Osborne. What is rotating in exploratory factor analysis. *Practical assessment, research & evaluation*, 20(2):1–7, 2015.

[28] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *ICLR*, 2016.

[29] Mengye Ren, Eleni Triantafillou, Sachin Ravi, Jake Snell, Kevin Swersky, Joshua Tenenbaum, Hugo Larochelle, and Richard Zemel. Meta-learning for semi-supervised few-shot classification. In *ICLR*, 2017.

[30] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *ICML*, 2015.

[31] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *ICML*, 2014.

[32] Karl Ridgeway and Michael C Mozer. Learning deep disentangled embeddings with the f-statistic loss. In *Advances in Neural Information Processing Systems*, pages 185–194, 2018.

[33] Alexander Shapiro. Identifiability of factor analysis: Some results and open problems. *Linear Algebra and its Applications*, 70:1–7, 1985.

[34] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In *NIPS*, 2017.

[35] Flood Sung, Yongxin Yang, Li Zhang, Tao Xiang, Torr. Philip, and Timothy Hospedales. Relation network for few-shot learning. In *CVPR*, 2018.

[36] Yuxing Tang, Josiah Wang, Boyang Gao, Emmanuel Dellandréa, Robert Gaizauskas, and Liming Chen. Large scale semi-supervised object detection using visual and semantic knowledge transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2119–2128, 2016.

[37] Jakub M Tomczak and Max Welling. Vae with a vampprior. *arXiv preprint arXiv:1705.07120*, 2017.

[38] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In *NIPS*, 2016.

[39] Carl Yang, Lanxiao Bai, Chao Zhang, Quan Yuan, and Jiawei Han. Bridging collaborative filtering and semi-supervised learning: a neural approach for poi recommendation. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1245–1254. ACM, 2017.

[40] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets. In *NIPS*, 2017.

[41] Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. Metagan: An adversarial approach to few-shot learning. In *NIPS*, 2018.