Max-margin learning with the Bayes factor

Rahul G. Krishnan
MIT

Arjun Khandelwal
MIT

Rajesh Ranganath
NYU

David Sontag
MIT

Abstract

We propose a new way to answer probabilistic queries that span multiple datapoints. We formalize reasoning about the similarity of different datapoints as the evaluation of the Bayes Factor within a hierarchical deep generative model that enforces a separation between the latent variables used for representation learning and those used for reasoning. Under this model, we derive an intuitive estimator for the Bayes Factor that represents similarity as the amount of overlap in representation space shared by different points. The estimator we derive relies on a query-conditional latent reasoning network, that parameterizes a distribution over the latent space of the deep generative model. The latent reasoning network is trained to amortize the posterior-predictive distribution under a hierarchical model using supervised data and a max-margin learning algorithm. We explore how the model may be used to focus the data variations captured in the latent space of the deep generative model and how this may be used to build new algorithms for few-shot learning.

1 INTRODUCTION

How do we frame the problem of selecting, from a target set, an object most similar to a given query set? For example—given a red chair, a blue chair and a black chair, we would rank chairs in the target set highly. At the same time, given a red chair, a red car and a red shirt, we would rank red objects highly. Between the two tasks, our understanding of the *data* has not changed; what has changed is our understanding of the *task based on the context* given by the query. The query highlights the relevant property of the data that is needed for solving a specific task. Such tasks appear in few-shot learning, where the goal is ranking objects according to their similarity to a given query set and in healthcare where a task may be finding similar patients to a given cohort.

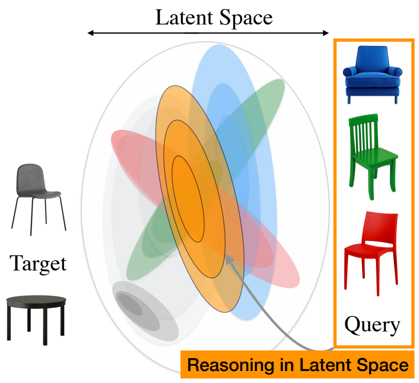


Figure 1: **Comparing objects in representational space:** On the left is a target set that will be ranked based on similarity to the query Q (right). The colour of each object is matched to a distribution in representation space. In orange is the output of the *latent reasoning network* – it represents the common factor of variation shared by $\mathcal{Q}$. The black chair should rank higher than the black table; here its distribution (in representation space) overlaps more with the output of the latent reasoning network.

To answer such queries, we could train discriminative models attuned to answering set-conditional queries at test time (e.g. Vinyals *et al.* (2016)). Or we could encode class separability in the structure of a generative model (Edwards & Storkey, 2017) and use inference for prediction. We take a different approach to the problem.

We learn a generic representation space (using unsupervised data) that is warped (using supervised data) for potentially different test-time problems. The task of scoring objects given a query is decomposed into two subtasks. The first determines the common property shared by items in the query set and represents the property as a region in representation space. In Figure 1, we visualize such a hypothetical space. On the right is a query comprising chairs of different colors and (in orange) a region of space that characterizes the property (in this case, a likeness to a chair) common to items in the query. The second task is to score a target item based on how much it expresses the region of representation space shared by items in the query. For the two candidate target points in Figure 1 (left), the black chair would rank rank highly since its representation has more in common with the property encapsulated by the query.

Here, we will use the latent space of deep generative models (Rezende *et al.* , 2014; Kingma & Welling, 2014) as our representation space. In such models, one can do posterior inference to map from raw data to a distribution in latent space. Then, to find commonalities among query items, we introduce a *latent reasoning network* (LRN). The LRN takes a query as input and constructs a probability distribution over the latent space that *summarizes* the representations of the query points into a single distribution. Figure 1 (orange) depicts what the output of the LRN might look like. We design a neural architecture for the LRN based on Zaheer *et al.* (2017) so that it does not dependent on the size of the query set. To score the latent space of a target item, we propose using the logarithm of the Bayes Factor (Jeffreys, 1998). The Bayes Factor measures how conditioning on the query alters the likelihood of a target point. Our approach is inspired by Bayesian Sets Ghahramani & Heller (2005) where data was assumed to be modeled by a hierarchical exponential family distribution and the likelihood ratio of the joint distribution and product of marginals was shown to be a useful measure of similarity.

The latent (representation) space of a deep generative models learned with unsupervised data is typically non-identifiable. i.e. there will exist multiple good (from the perspective of log-likelihood) representation spaces. Each corresponds to a different notion of similarity and a different way of grouping points. However, queries provide extra information: they reveal which points should be close together in latent space. We take advantage of this and propose a supervised max-margin learning algorithm for the LRN such that scores given to items in the query are larger than scores unrelated to the query.

We obtain a coupled set of models: in which one model is a deep generative model of the data whilst the other reshapes the latent space of the first and serves to answer queries about similarity judgements between datapoints. We study how the proposed approach can tune the latent space of deep generative models and be used to build new types of models for few-shot learning. We begin in Section 2 by motivating the Bayes Factor as a viable tool for computing similarity.

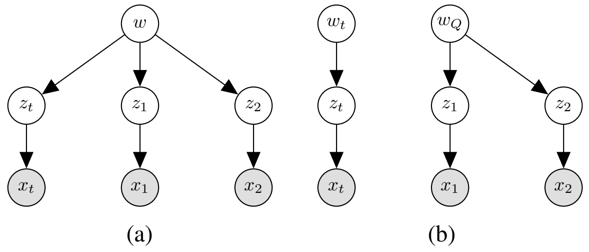


Figure 2: **Hypothesis testing with Deep Generative Models:** (a) The **Reasoning Model**, here, depicting the hypothesis that the set $\{x_t, \mathcal{Q} = \{x_1, x_2\}\}$ was generated jointly; (b) the two figures represent the hypothesis that x_t and $\mathcal{Q}$ were generated independently under different realizations of w (the random variable that captures the property shared across datapoints).

2 FROM REPRESENTATION LEARNING TO REASONING

Here, we consider the problem of scoring elements in a set based on how similar they are to a given query. Suppose we are given a dataset $\mathcal{D} = \{x_1, \ldots, x_N\},\ x_i \in \mathbb{R}^n, x_i \in \mathcal{D}$. Then for a query $\mathcal{Q} = \{x_1, \ldots, x_Q\};\ |\mathcal{Q}| = Q$, we wish to assign to each $x_t \in \mathcal{D}$ a score$(x_t, \mathcal{Q})$ that denotes how *similar* x_t is to elements of the query $\mathcal{Q}$.

2.1 The Data Model

A simple way to quantify how similar objects are (here, between $\mathcal{Q}$ and x_t) might be to take the pairwise Euclidian distance between them. For complex, high dimensional data that do not lie on a Euclidian manifold, such a metric may fail to capture interesting regularity between data.

Alternatively, we can use a latent variable model to construct a representation of data. The latent variable then becomes a low-dimensional sufficient statistic the raw data when quantifying similarity. The simplest latent variable model we will consider has the following generative process: $z \sim p_{\textbf{dm}}(z);\ x \sim p_{\textbf{dm}}(x; f(z;\theta))$ where $p_{\textbf{dm}}(z)$ is a simple distribution such as $\mathcal{N}(0, I)$. The use of MLPs in the conditional distributions allow the model to fit highly complex data despite the use of a simple prior. When f is parameterized by a Multi-Layer Perceptron (MLP), the resulting model is a deep generative model. We will refer to this model (Kingma & Welling, 2014; Rezende *et al.* , 2014) as the **Data Model** (with probabilities denoted with subscript $_{\textbf{dm}}$).

The generative process assumes datapoints are drawn independently. Using variational inference with an inference network (Hinton *et al.* , 1995) to approximate the posterior distribution, $p_{\textbf{rm}}(z|x)$, the model can be learned by maximizing a lower bound on the log-likelihood of the data obtained using Jensen's inequality:

$$\log p_{\mathbf{dm}}(x;\theta) \geq \mathop{\mathbb{E}}_{q_{\mathbf{dm}}(z|x;\phi)} [\log p_{\mathbf{dm}}(x|z;\theta))] \\ - \mathrm{KL}(\, q_{\mathbf{dm}}(z|x;\phi)||p_{\mathbf{dm}}(z)\,) = \mathcal{L}(x;\theta,\phi), \quad (1)$$

With a Gaussian distribution as the variational approximation: $q_{\mathbf{dm}}(z|x;\phi) \sim \mathcal{N}(\mu_\phi(x), \Sigma_\phi(x))$ where $\mu_\phi(x), \Sigma_\phi(x)$ are (diferentiable, parameteric, with parameters ϕ) functions of the observation x. Eq. 1 is differentiable in θ, ϕ (Kingma & Welling, 2014; Rezende *et al.* , 2014) and the model parameters (θ, ϕ) can be learned via gradient ascent on $\mathcal{L}(x;\theta,\phi)$.

With the variational approximation, $q_{\mathbf{dm}}(z|x;\phi)$, to map from data to latent space, would computing overlap in the posterior distributions of points in $\mathcal{Q}$ and x_t suffice to identify similar points? The answer is *sometimes*. While unsupervised learning will tend to put *similar* points together, the notion of similarity encoded in the latent space need not correspond to the notion of similarity required for a task at test time. We require a way to guide the structure of the latent space to be better suited for a task.

2.2 The Reasoning Model

Introducing hierarchy into the generative process is one way to guide the structure of latent variables. In Figure 2 (b) is a simple hierarchical model that makes explicit the insight that similar datapoints should have similar latent spaces. It defines the following generative process for a set of similar objects $\mathcal{Q}$: $p_{\mathbf{rm}}(\mathcal{Q}) = \int_w \int_z p_{\mathbf{rm}}(w) \prod_{q=1}^{Q} p_{\mathbf{rm}}(z_q|w) p_{\mathbf{rm}}(x_q|z_q)$. The random variable w defines the context of $\mathcal{Q}$. It may denote the label or class identity of points in $\mathcal{Q}$ but more broadly is a representation of the properties that points in $\mathcal{Q}$ satisfy. For notational convenience and because we can express reasoning about similarity as a probabilistic query in this model, we refer to it as the **Reasoning Model**.

The Neural Statistician (Edwards & Storkey, 2017) uses $\mathrm{KL}(p(w|x_t)||p(w|\mathcal{Q}))$ to quantify the similarity between x_t and $\mathcal{Q}$ in a model similar to the one in Figure 2 (b). In this work, we pose the estimation of similarity between objects as hypothesis testing in a hierarchical deep generative model. The conditional independences in Figure 2 (b) enforce that x_t is independent of w given z_t, i.e. the per-data-point latent variables serve as a sufficient statistic to quantify comparisons between multiple datapoints. The conditional density $p(x_t|z_t)$ is a map from the representation space to the data while $p(z_t|w)$ dictates how the latent space of a datapoint behaves as a function of property encoded in w.

2.3 Bayes Factor

To score the similarity between two objects (in this case x_t and set $\mathcal{Q}$) under the Reasoning Model, we turn to the likelihood ratio between the joint distribution of x_t and $\mathcal{Q}$ and the product of their marginals. If x_t and $\mathcal{Q}$ are drawn from the same joint distribution, then there exists a random variable w that governs the distribution of the *latent* spaces $z_t, z_1, \ldots, z_Q$. With slight abuse of notation [1], Figure 2 (a) depicts this scenario when $\mathcal{Q} = \{x_1, x_2\}$. If x_t and $\mathcal{Q}$ are not similar, then their latent spaces will have different distributions, and they are children of different realizations of w (see Figure 2 (b)). With that in mind, the score function we use to measure similarity is given by (**Bayes Factor**):

$$\frac{p(x_t, \mathcal{Q})}{p(x_t)p(\mathcal{Q})} = \frac{p(x_t|\mathcal{Q})}{p(x_t)} = \mathrm{score}(x_t, \mathcal{Q}) \quad (2)$$

The log-score is the pointwise mutual information (Fano, 1949), a measure of association that is frequently used in applications such as natural language processing (Church & Hanks, 1990). The Bayes Factor normalizes the posterior predictive density of the target point conditioned on the query by the target's marginal likelihood under the model. It also has an information theoretic interpretation. Letting $h(x) = -\log p(x)$ denote the self-information (or surprisal), then $\log \mathrm{score}(x_t, \mathcal{Q}) = h(x_t) - h(x_t|\mathcal{Q})$ intuitively denotes the surprise (quantified in nats or bits) from observing x_t when having already observed $\mathcal{Q}$.

Similarity in Latent Space: Equation 2 captures an intuitive notion of similarity but evaluating $p(x_t)$, the marginal density of the target, is typically intractable (except in hierarchical models that lie in the exponential family (Ghahramani & Heller, 2005)). Furthermore, an importance sampling based Monte-Carlo estimator for $p(x_t)$ will involve a high-dimensional integral in the data x_t. We therefore propose the following decomposition of the score function that evaluates the Bayes Factor in the target datapoint's (lower dimensional) latent space:

$$\begin{aligned} \frac{p_{\mathbf{rm}}(x_t|\mathcal{Q})}{p_{\mathbf{rm}}(x_t)} &= \frac{1}{p_{\mathbf{rm}}(x_t)} \int_{z_t} p_{\mathbf{rm}}(x_t, z_t|\mathcal{Q}) \quad (3) \\ &= \frac{1}{p_{\mathbf{rm}}(x_t)} \int_{z_t} p_{\mathbf{rm}}(x_t|z_t) p_{\mathbf{rm}}(z_t|\mathcal{Q}) \\ &= \frac{1}{p_{\mathbf{rm}}(x_t)} \int_{z_t} \frac{p_{\mathbf{rm}}(z_t|x_t) p_{\mathbf{rm}}(x_t)}{p_{\mathbf{rm}}(z_t)} p_{\mathbf{rm}}(z_t|\mathcal{Q}) \\ &= \int_{z_t} \underbrace{\frac{p_{\mathbf{rm}}(z_t|x_t)}{p_{\mathbf{rm}}(z_t)}}_{\textbf{Relative Posterior Likelihood}} \underbrace{p_{\mathbf{rm}}(z_t|\mathcal{Q})}_{\textbf{Latent Reasoning Network}} . \end{aligned}$$

[1] We re-use Figure 2 to denote both the instantiation of a hypothesis and the generative process

The estimator above formalizes the intuition for comparing points laid out in Section 1. The query-conditional posterior-predictive density over the latent space of the target datapoint, $p_{\textbf{rm}}(z_t|\mathcal{Q})$, reasons about points in the query and represents them as a density in latent space, The **Relative Posterior Likelihood**, $\frac{p_{\textbf{rm}}(z_t|x_t)}{p_{\textbf{rm}}(z_t)}$ scores how likely the target point is to have come from the relevant part of latent space.

3 HIERARCHICAL MODELS WITH COMPOUND PRIORS

To compute the ratio $\frac{p_{\textbf{rm}}(z_t|x_t)}{p_{\textbf{rm}}(z_t)}$, we need to marginalize w_t. However, under certain assumptions about the conditional distributions in the **Reasoning Model**, we will see that approximating this ratio becomes simpler.

Assumption 1. Priors with Compound Distributions

$$\int_w p_{\textbf{rm}}(w) p_{\textbf{rm}}(z|w) dw = p_{\textbf{dm}}(z)$$

Assumption 2. Matching conditional likelihoods

$$p_{\textbf{rm}}(x|z) = p_{\textbf{dm}}(x|z)$$

Lemma 1. *Matching posterior marginals*

$$p_{\textbf{dm}}(z|x) = p_{\textbf{rm}}(z|x)$$

Proof. Follows from Bayes rule and Assumption 1, 2. □

Lemma 2. *Matching marginal likelihoods*

Under Assumption 1 and 2:

$$p_{\textbf{dm}}(x) = p_{\textbf{rm}}(x)$$

Proof.

$$\begin{aligned} p_{\textbf{rm}}(x) &= \int_w \int_z p_{\textbf{rm}}(w) p_{\textbf{rm}}(z|w) p_{\textbf{rm}}(x|z)] dz dw \\ &= \int_z p_{\textbf{dm}}(z) p_{\textbf{dm}}(x|z) dz = p_{\textbf{dm}}(x) \end{aligned}$$

□

The conditions above state when we can take an instance of the **Data Model** discussed in Section 2.1 and transform it into an instance of the **Reasoning Model** in Section 2.2 while preserving the marginal likelihood of the data.

This transformation has a few implications. The first is when evaluating the Bayes Factor; if we work in a class of **Reasoning Models** that satisfy Assumption 1, then we can evaluate the **Relative Posterior Likelihood** using the prior and posterior distribution of the associated **Data Model**. With Lemma 1 and Assumption 1:

$$\frac{p_{\textbf{rm}}(x_t|\mathcal{Q})}{p_{\textbf{rm}}(x_t)} = \int_{z_t} \underbrace{\frac{p_{\textbf{dm}}(z_t|x_t)}{p_{\textbf{dm}}(z_t)}}_{\textbf{Relative Posterior Likelihood}} \underbrace{p_{\textbf{rm}}(z_t|\mathcal{Q})}_{\textbf{Latent Reasoning Network}}$$

where $p_{\textbf{dm}}(z_t)$ is typically fixed ahead of time (e.g. $\mathcal{N}(0;\mathbb{I})$) and we can do inference for $p_{\textbf{dm}}(z_t|x_t)$ (or approximate it using the inference network $q_{\textbf{dm}}(z|x;\phi)$).

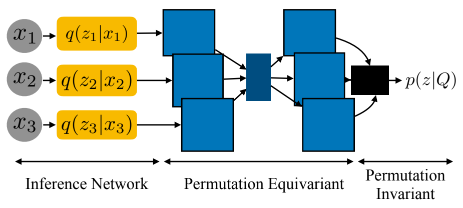
The second implication is that part of the **Reasoning Model**, $p_{\textbf{rm}}(x|z)$, can be learned ahead of time. This gives us the flexibility to *warm-start* the **Reasoning Model** using a *pre-trained* **Data Model** whose $p_{\textbf{dm}}(z)$ can be expressed according to Assumption 1. In this way, even if we do not know which property will be used to organize datapoints into sets at test time, we can still learn a generic low-dimensional representation of the dataset. We will make use of this when we discuss the learning framework in Section 5. For now, what remains is how we can specify $p_{\textbf{rm}}(w), p_{\textbf{rm}}(z|w)$ in order to evaluate $p_{\textbf{rm}}(z_t|\mathcal{Q})$.

4 LATENT REASONING NETWORKS

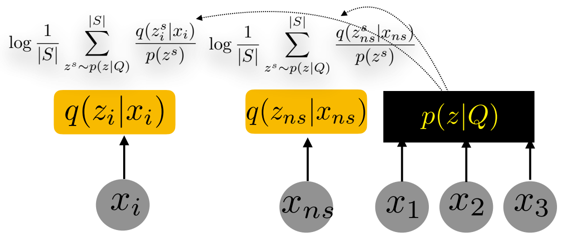
Although $p_{\textbf{rm}}(z_t|\mathcal{Q}) = \int_w p_{\textbf{rm}}(z_t|w), p_{\textbf{rm}}(w|\mathcal{Q}) dw$, finding $p_{\textbf{rm}}(w), p_{\textbf{rm}}(z|w)$ that satisfy Assumption 1 may prove challenging and so we will make use of another computational trick. To evaluate the Bayes Factor we only need a way to sample from $p_{\textbf{rm}}(z_t|\mathcal{Q})$ i.e. the posterior predictive distribution given the query, of the target's latent representation. Our strategy therefore, will instead be to parameterize and learn $p_{\textbf{rm}}(z_t|\mathcal{Q})$ directly from data.

Without $p_{\textbf{rm}}(w), p_{\textbf{rm}}(z|w)$, we lose the ability to sample from the Reasoning Model but by amortizing $p_{\textbf{rm}}(z_t|\mathcal{Q})$ we obtain a fast way to evaluate the Bayes Factor at test time. $p_{\textbf{rm}}(z_t|\mathcal{Q})$ must *reason* about how the latent spaces of points in $\mathcal{Q}$ are related and parameterize a distribution over the latent space of the target datapoint x_t; this distribution must characterize the property represented by points in $\mathcal{Q}$. Therefore, we refer to this amortizated, parameteric posterior-predictive distribution as a *Latent Reasoning Network.* Since we do not know the functional form of this distribution we will parameterize it as a non-linear function of the query $\mathcal{Q}$.

To construct the LRN, we require neural architectures capable of operating over sets. We make use of two primitives for such neural architectures proposed by Zaheer *et al.* (2017). These functions operate over sets of vectors $\mathcal{Q} = \{x_1, \ldots, x_Q\}, x_q \in \mathbb{R}^n$. We will use the notation $\mathbb{R}^{n\times|\mathcal{Q}|}$ to denote a set of size $|\mathcal{Q}|$ where each element is an n-dimensional vector. We design the LRN, with the following three properties:



(a) **Latent Reasoning Network**


$\log \frac{1}{|S|} \sum_{z^s \sim p(z|Q)}^{|S|} \frac{q(z_i^s|x_i)}{p(z^s)}$
$\log \frac{1}{|S|} \sum_{z^s \sim p(z|Q)}^{|S|} \frac{q(z_{ns}^s|x_{ns})}{p(z^s)}$
$q(z_i|x_i)$
$q(z_{ns}|x_{ns})$
$p(z|Q)$
x_i
x_{ns}
x_1
x_2
x_3

(b) **Loss function**

Figure 3: **Latent Reasoning Networks (LRN) and Loss function:** On the left is a diagrammatic representation of $p_{\textbf{rm}}(z_t|\mathcal{Q})$. On the right is a depiction of Monte-Carlo sampling (with samples from the LRN) to evaluate Bayes factor. x_i is a point similar to those in the query $\mathcal{Q} = \{x_1, x_2, x_3\}$, while x_{ns} is not. We suppress subscripts in the figure.

A] Parameter Sharing: We share parameters between the inference network of the Data Model and the LRN. A direct consequence of this choice is that the LRN now has the ability to change the way inference is done in the Data model. The first stage of the LRN uses the inference network of the Data Model to map from the set $\mathcal{Q}$ to a set of each point's variational parameters

B] Exchangeability: The output of the LRN must not depend on the order of elements in $\mathcal{Q}$. We achieve this by using the functions proposed by (Zaheer *et al.* , 2017): $g : \mathbb{R}^{n\times|\mathcal{Q}|} \rightarrow \mathbb{R}^{m\times|\mathcal{Q}|}$ is a permutation equivariant function that maps from sets of n dimensional vectors to sets of m dimensional vectors while ensuring that if the input elements were permuted, then the output elements would also be permuted identically. The form of g is given by $g(\mathcal{Q}) = \left[\rho\left(W_1^{\text{eq}} x_q + W_2^{\text{eq}}(\sum_{q'} x_{q'})\right)\right]_{q=1}^{|\mathcal{Q}|}$ where $W_1^{\text{eq}} \in \mathbb{R}^{m\times n}$, $W_2^{\text{eq}} \in \mathbb{R}^{m\times n}$ and ρ is an elementwise nonlinearity. We use compositions of the function g in the second stage of the LRN to *learn* about how the variational parameters between points in $\mathcal{Q}$ relate to one-another and map to a set of intermediate representations.

C] Distributions in latent space: The network must parameterize a valid density in latent space; this is satisfied by construction. To go from the set of intermediate representations to the parameters of $p(z_t|\mathcal{Q})$, we leverage the following permutation invariant function: $f(\mathcal{Q}) = \rho\left(\sum_q (W^{\text{inv}} x_q + b)\right)$, $f : \mathbb{R}^{n\times|\mathcal{Q}|} \rightarrow \mathbb{R}^m$ where $W^{\text{inv}} \in \mathbb{R}^{m\times n}, b \in \mathbb{R}^m$ are linear operators and ρ is an elementwise non-linearity.

With $\mu(\mathcal{Q};\gamma,\phi), \Sigma(\mathcal{Q};\gamma,\phi)$ as parametric functions of set $\mathcal{Q}$, we can write $p_{\textbf{rm}}(z_t|\mathcal{Q};\gamma,\phi) = \mathcal{N}(\mu(\mathcal{Q};\gamma,\phi), \Sigma(\mathcal{Q};\gamma,\phi))$. γ denotes the parameters of the permutation equivariant and invariant layers while ϕ represent the parameters shared with $q_{\textbf{dm}}(z|x;\phi)$. We visualize the LRN in Figure 3a.

5 LEARNING

The learning procedure we use is based on a combination of doing unsupervised learning to learn a good representation alongside a supervised max-margin loss to ground the representation for a specific task. We discuss each separately and then highlight how they are combined.

Unsupervised Learning: Since we use **Reasoning Models** that satisfy Assumption 1, 2, we make use of the transformation between the **Data Model** and **Reasoning Model** in Section 3. We maximize the likelihood of a given dataset using the lower-bound in Equation 1. A consequence of doing variational learning of the Data Model is that we can use $q_{\textbf{dm}}(z|x;\phi)$ to approximate the Bayes Factor.

Max-Margin Learning: We expect that the Bayes Factor in Equation 3 takes a high value when the target point x_t is *similar* to $\mathcal{Q}$ and a low value when x_t is dissimilar to $\mathcal{Q}$. But how do we know what points form $\mathcal{Q}$? This will depend on the test-time task. We assume we are given labels that define the property encompassed in sets of datapoints.

Assumption 3. For L datapoints in $\mathcal{D}$, we have $\mathcal{Y} = \{y_{x_1}, \ldots, y_{x_L}\}$, $y_l \in \{1, \ldots, K\}$ where y_{x_i} is the label for x_i that takes one of K unique labels. We define $\mathbb{N}_{x_i}^{\mathcal{Q}} = \{x_k \textit{ s.t. } y_{x_k} \in \mathcal{Y} \;\&\; y_{x_k} = y_{x_i}\}$, $\mathbb{N}_{x_i}^{\not{\mathcal{Q}}} = \{x_k \textit{ s.t. } y_{x_k} \in \mathcal{Y} \;\&\; y_{x_k} \neq y_{x_i}\}$ to be sets of datapoints that have the same label as x_i and those that do not.

We will assume that a point can only have a single label. Here, the labels characterize the property we want to base our similarity judgements on. Therefore, learn the parameters of $p(z_t|\mathcal{Q};\gamma,\phi)$ using the following (supervised) loss function:

$$\mathcal{L}_{\textbf{mm}}(x;\gamma,\phi) = \mathbb{E}_{\mathcal{Q}\sim\mathbb{N}_x^{\mathcal{Q}}} \mathbb{E}_{\mathcal{Q}_{ns}\sim\mathbb{N}_x^{\not{\mathcal{Q}}}} \frac{1}{|\mathcal{Q}_{ns}|} \sum_{x_{ns}\in\mathcal{Q}_{ns}} \max(\log \text{score}(x_{ns},\mathcal{Q}) - \log \text{score}(x,\mathcal{Q}) + \Delta, 0). \quad (4)$$

The loss function maximizes the difference between the log-Bayes Factor for points that lie within the set $\mathcal{Q}$ and those that do not (they lie in $\mathcal{Q}_{ns}$). The $\log \text{score}(x, \mathcal{Q})$, in Equation 3, is evaluated via Monte-Carlo sampling and the log-sum-exp trick. The expectation is differentiable with respect to γ, ϕ via the reparameterization trick (Kingma & Welling, 2014; Rezende *et al.* , 2014). For the margin Δ we use the mean-squared-error between the the posterior means of x, x_{ns}. We provide a visual depiction of how the loss is evaluated using the LRN in Figure 3.

Combined Loss: With the unsupervised learning objective for the **Data Model** and the supervised max-margin loss function (Equation 4) for the LRN, we obtain the following loss to jointly learn θ, ϕ, γ:

$$\min_{\theta.\phi.\gamma} \frac{1}{N} \sum_{i=1}^{N} \frac{1}{C+1} \left[-\mathcal{L}(x_i; \theta, \phi)\right] + \frac{C}{C+1} \mathbb{I}[x_i \in \mathcal{Y}] \mathcal{L}_{\textbf{mm}}(x_i; \gamma, \phi) \tag{5}$$

where C is a regularization constant that trades off between the supervised and the unsupervised loss. The unsupervised loss learns a representation space constrained to lie close to the prior while explaining the data under the generative model. The max-margin loss modifies this representation space so that dissimilar points are kept apart. Note that Equation 5 is no longer a valid bound on the marginal likelihood of the training set (for $C > 0$).

6 EVALUATION

The goal of this section is threefold: (1) to study whether $p_{\textbf{rm}}(z|\mathcal{Q})$is learnable from data using the max-margin learning objective–we expect this to be challenging since we learn the parameters of a model that is itself used to evaluate the the score function in the loss; (2) studying the role of parameter sharing between the inference network and the LRN – i.e. whether the latter can change the former in adversarial scenarios; and (3) studying the utility of the framework for few-shot learning.

We will release code in Keras (Chollet *et al.* , 2015). The supplementary material contains detailed information on the neural architectures of the deep generative models used in the evaluation. We learn parameters with a learning rate of 0.00005 and adaptive momentum updates given by ADAM (Kingma & Ba, 2015). We set the value C separately for each experiment. When there is a task to be solved, C can be set using the validation data. When using a pre-trained Data Model, we found it useful to anneal C from a higher to a lower value so that the task-specific supervised term can overcome (potentially) suboptimal latent spaces learned from unsupervised data. We use the following datasets for our study:

Synthetic Pinwheel: A synthetic dataset of two-dimensional points arranged on a pinwheel taken from the work of Johnson *et al.* (2016). We depict the raw data in Figure 4a. The dataset is created with five labels.

MNIST digits: 50000 black and white images of handwritten digits (LeCun *et al.* , 1998).

MiniImagenet: A subsampled set of images taken from the Imagenet repository setup for the task of k-shot learning by Vinyals *et al.* (2016). We use the train-validate-test split kindly provided by Ravi & Larochelle (2016).

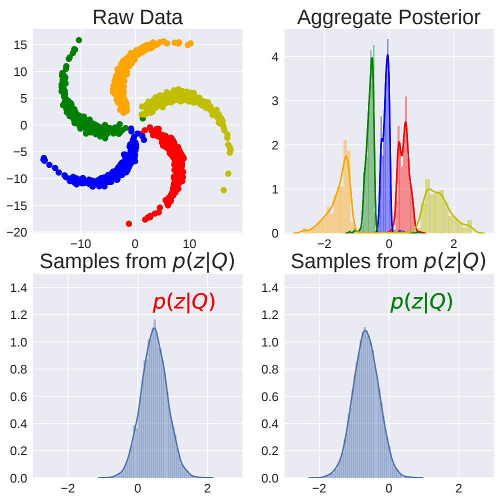
6.1 Learning $p(z|\mathcal{Q})$

As a sanity check, we begin by first training a deep generative model (without labels and using a one-dimensional latent space) on the Pinwheel dataset. We visualize the raw-data and learned aggregate posterior $\sum_x q_{\textbf{dm}}(z|x; \phi)$ in Figure 4a (top row). We see that the unsupervised learning alone induces class separation in the aggregate posterior distribution. Using the learned model, we hold fixed parameters: θ, ϕ and learn the parameters γ of the LRN using the loss function in 4 with $C = 2000$. We form a kernel density estimate of samples from $p_{\textbf{rm}}(z|\mathcal{Q})$ using randomly constructed sets of points derived from the red and green clusters. In Figure 4a (bottom row), we see that samples from the LRN correspond to regions of the latent space associated with $\mathcal{Q}$. On synthetic examples, the LRN finds regions of latent space corresponding to points from a query $\mathcal{Q}$.

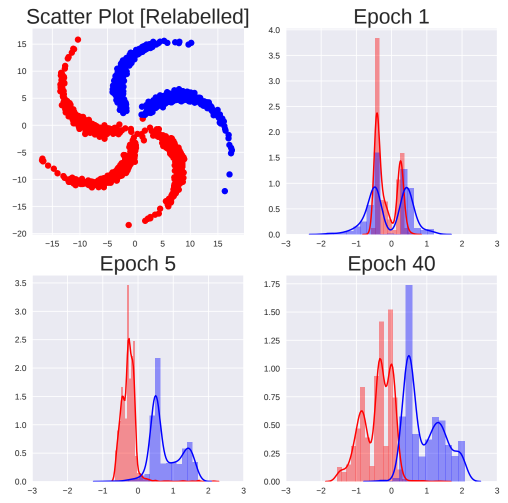
6.2 Changing inductive biases at test-time

Previously, we worked with a model where the structure of the latent space (as seen in the aggregate posterior distribution) formed during unsupervised learning co-incided with how points were grouped into sets. Here, we study what happens where the notion of which points are similar changes at test time. We relabel the pinwheel dataset so that the yellow and orange points form one class while the green, red and blue form the other (see Figure 4b, top left). This corresponds to an *adversarial* labelling of the data since we use a deep generative model in which points in the same class are far apart in the learned latent space. If we keep θ, ϕ fixed then $p_{\textbf{rm}}(z|\mathcal{Q})$ (whose output is parameterized as a unimodal Gaussian distribution) cannot capture the relevant subspace.

We have two choices here; we can either consider richer parameterizations for $p_{\textbf{rm}}(z|\mathcal{Q})$ that are capable of capturing multi-modal structure in the latent space using techniques proposed by (Rezende & Mohamed, 2015), or we



(a) **Data and Aggregate Posterior:** (Top Left) Raw Pinwheel Data; (Top Right) Aggregate posterior density of a learned (unconditional) deep generative model coloured by class membership. (Bottom Row) Sampling from $p_{\text{rm}}(z|\mathcal{Q})$ where the colour denotes the class membership of points in $\mathcal{Q}$.



(b) **Learning dynamics:** (Top left) Visualization of *adversarially labelled data* (relative to the learned aggregate posterior in Figure 4a (top right)). The remaining plots are class coloured visualizations of the aggregate posterior (during training) while allowing the LRN to fine-tune the latent space of the DGM.

Figure 4: **Qualitative Evaluation on Pinwheel Data**

can instead allow the $p_{\text{rm}}(z|\mathcal{Q})$ to *change* the underlying latent space of the generative model by back-propagating through the parameters of the inference network. Here, we opt for the latter though the former is an avenue for future work.

We minimize Equation 5 while annealing the constant C from $1000 \rightarrow 1$ linearly through the course of training. To gain insight into the learning dynamics of the LRN during training, we visualize the aggregate posterior of the generative model (via the fine-tuned inference network) in Figure 4b through the course of training. The role of this adversarial scenario is to highlight two important points (1) unsupervised learning is typically unidentifiable and may not learn a representation appropriate to all tasks and (2) learning with the latent reasoning network can overcome a suboptimal (relative to the task at hand) representation and transform it to a more suitable one.

6.3 Modeling High Dimensional Data

Inducing diversity in latent space: Moving beyond low-dimensional data, we study learning LRNs on MNIST digits. We use a Data Model with a two-dimensional latent space for this experiment. We begin by training the model in a fully unsupervised manner and visualize the learned latent space in the form of the aggregate posterior (Figure 5a [left]). Although there is some class separability, we find that the unsupervised learning algorithm concentrates much of the probability mass together.

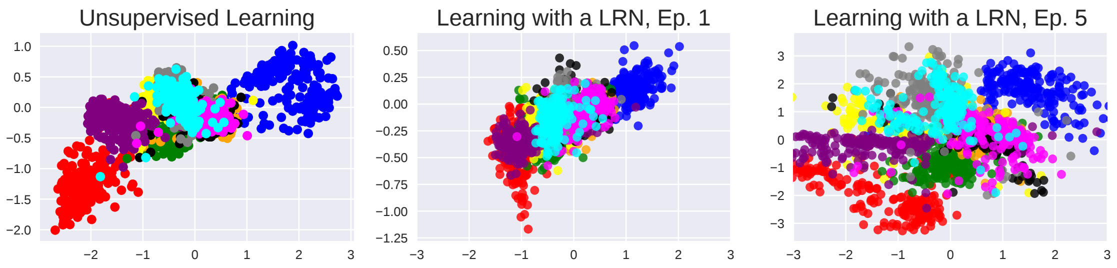
We re-learn the same model with the loss in Equation 5 where C is set to 3000 (and annealed to 1). We again visualize the new aggregate posterior distribution of the Data Model in Figure 5a (middle and right). When learning with Equation 5, the inference network uses more of the latent space in the model because the max-margin loss pushes points in different classes further apart.

Qualitative Analysis of MNIST digits: To validate our method, we provide visualizations on the MNIST dataset. We select a handful of labelled examples $\mathcal{Q}$ (Figure 5b, left) and visualize both their posterior means and samples from $p(z|\mathcal{Q})$ (Figure 5b, middle). Then, for each sample from $p_{\text{rm}}(z|\mathcal{Q})$, we evaluate the fine-tuned $p_{\text{dm}}(x|z)$ and visualize the images in Figure 5b (right). We see that the generative model fine-tuned with the learning algorithm retains its ability to generate meaningful samples.

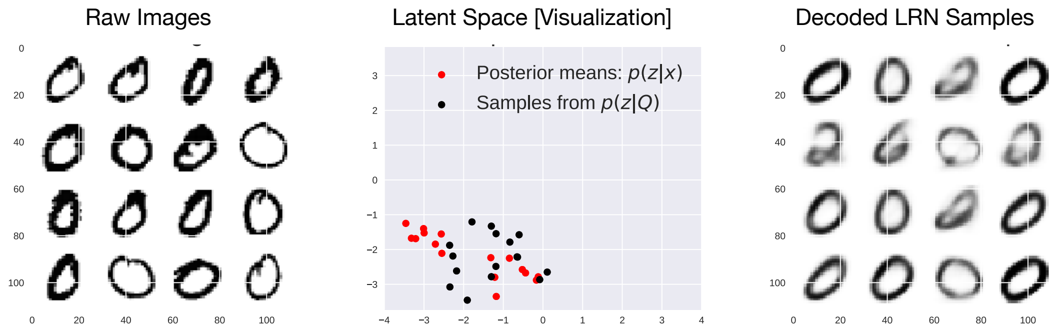
6.4 Few-shot learning with the Bayes Factor

The task of k-shot learning is to identify the class an object came from given a single example from 5 other classes (1-shot, 5-way). In the 5-shot, 5-way task. there are 5 examples provided from each of the 5 potential classes. We use an LRN with a deep-discriminative model to obtain near state of the art performance in few-shot learning on the MiniImagenet dataset.

Following (Bauer *et al.* , 2017), who show that discriminative models alone form powerful baselines for this task



(a) **Training dynamics for MNIST:** Aggregate (two-dimensional) posterior of deep generative model of MNIST (coloured by label). The left corresponds to a model trained with unsupervised data only; the middle & right show the aggregate posteriors for a model fine-tuned using Equation 5.



(b) **Test-time evaluation of LRN on MNIST:** On the left are a set of query points $\mathcal{Q}$ drawn from the same class, in the middle, we visualize samples from $q_{\textbf{dm}}(z|x;\phi)$ for each of the points and $p_{\textbf{rm}}(z|\mathcal{Q})$. On the right is the output of the fine-tuned conditional density $p_{\textbf{dm}}(x|z)$ for samples drawn from $p_{\textbf{rm}}(z|\mathcal{Q})$.

Figure 5: **Qualitative Evaluation on MNIST**

on this dataset, we pretrain an 18 layer Resnet (He *et al.* , 2016) convolutional neural network to predict class labels at training time. We use early stopping on a validation set based on the nearest neighbor performance of the learned embeddings (obtained from the final layer of the ResNet) to identify the best model. Building a good generative model of the images in MiniImagenet is difficult and so instead, we use the *fixed* embeddings as a 256 dimensional proxy for each image. We initialize $q_{\textbf{dm}}(z|x;\phi)$ with the pretrained Resnet and set up a deep generative model to maximize the likelihood of the fixed embeddings (after discriminative pre-training).

For this task, when comparing to the many different approaches proposed, it is challenging to control for both the depth of the encoder that parameterizes the representation and the various algorithmic approach used to tackle the problem using the representation. Therefore, our two take-aways from Table 1 are: (1) on the 1 shot and 5 shot task, we outperform a strong nearest neighbors baseline created using fixed (but learned) embeddings suggesting that our algorithmic approach bears promise for this task and (2) the method is competitive with other state of the art approaches.

Table 1: Accuracy on the 5-way MiniImagenet task

MODEL	1-SHOT	5-SHOT
NEAREST NEIGHBOR	51.4 ± 0.08	67.5 ± 0.08
OURS [RESNET18 ENCODER]	53.5 ± 0.08	68.8 ± 0.08
MATCHING NETWORKS (VINYALS *et al.* , 2016)	46.6	60.0
MAML (FINN *et al.* , 2017)	48.7	63.1
PROTOTYPICAL NETS (SNELL *et al.* , 2017)	49.4	68.2
METANETS (MUNKHDALAI & YU, 2017)	49.2	*
TCML (MISHRA *et al.* , 2018)	56.7	68.9

7 RELATED WORK

Max Margin Learning: Max margin parameter estimation has been widely used in machine learning (e.g. in structural SVMs (Yu & Joachims, 2009) and in discriminative Markov networks (Zhu & Xing, 2009)). (Li *et al.* , 2015) give a doubly stochastic subgradient algorithm for regularized maximum likelihood estimation when dealing with max-margin posterior constraints.

(Zaheer *et al.* , 2017) experiment with max-margin learning using a variant of the DeepSets model to predict a scalar score conditioned on a set. While (Zaheer *et al.* , 2017) cite the estimator in (Ghahramani & Heller, 2005) as motivation for their model, they do not explicitly use, parameterize, or differentiate through the Bayes Factor in a *generative* model of data.

Inductive Transfer and Metric Learning: Lake *et al.* (2013) use probabilistic inference in a hierarchical model to classify unseen examples by their probability of being in a new class. Instead of the Bayes Factor, they use the posterior predictive obtained via the use of a MCMC algorithm to score target points relative to a query. (Ghahramani & Heller, 2005) evaluate the Bayes factor analytically in exponential family distributions. What we gain in for sacrificing tractability is the ability to work within a richer class of models. Though not motivated within the context of a hierarchical model, (Engel *et al.* , 2018) use an adversarial loss to recognize regions of latent space that correspond to points with a specified class.

Vinyals *et al.* (2016) learn a parametric K-nearest neighbor classifiers to predict whether a target item is within the same class as k-others. (Snell *et al.* , 2017) associate a point with a prototype within a set and use it to answer whether an object is in the same class as others. The Neural Statistician (Edwards & Storkey, 2017) learns a model similar [2] to the **Reasoning Model** in Figure 2 (b) by maximizing the likelihood of sets $\mathcal{Q}$. Their method does not use the Bayes Factor to score items; it also does not permit easy initialization with pre-trained Data Models since the full model is trained with queries.

We tune the latent space of a deep generative model to enhance class separability for test time tasks. By contrast, meta learning algorithms learn to tune the parameters of an algorithm or a model. (Finn *et al.* , 2017) prime the parameters of a neural network to have high accuracy at test time using second order gradient information.

Our work has close parallels with metric-learning; here the metric learned lies in the latent space of a deep generative model. (Bar-Hillel *et al.* , 2005) proposed Relevant Component Analysis, an optimization problem that jointly performs (linear) dimensionality reduction and learns a Mahalanobis metric using queries.

8 DISCUSSION

We seek good, task-specific inductive biases to quantify how similar a point is to a set. We give new theoretical and practical constructs towards this goal. We break up the problem into two parts: learn a good representation and tune the learned representation for a specific notion of similarity. Using the latent space in a deep generative model as our representation, we use the Bayes Factor to quantify similarity.

We derive conditions under which there exists an equivalence between a generative model where data are generated independently to a hierarchical model that jointly generates sets of (similar) points. Using this insight, we derive an easy-to-evaluate estimator for the Bayes Factor; the estimator poses the comparison between a point and a set as overlap in latent space. With the Bayes Factor as a differentiable scoring mechanism, we give a max-margin learning algorithm capable of changing the inductive bias of a (potentially pre-trained) deep generative model. To evaluate the Bayes Factor, we propose a neural architecture for a *latent reasoning network*: a set conditional density that amortizes the posterior predictive distribution of a hierarchical model.

Our approach has limitations. By directly parameterizing the posterior predictive density, and not the prior $p_{\textbf{rm}}(w)$ and conditional $p_{\textbf{rm}}(z|w)$, we lose the ability to sample points from the hierarchical generative model. Working with a set of models in which Assumption 1 holds may implicitly only find posterior predictive densities under relatively simple model families of $p_{\textbf{rm}}(w)$ and $p_{\textbf{rm}}(z|w)$. Finally, enforcing that property identity in w is conditionally independent of the data x, given the representation z, may make for a challenging learning problem – z has to represent both the property and variability in the property conditional distribution of the data.

An avenue of future work is leveraging vast amounts of unlabeled data for representation learning informed by a small amount of supervision to guide either during learning, or after learning, the structured of the learned space. Yet another interesting direction would be to learn LRNs that parameterize distributions over multiple, per-data-point latent variables.

ACKNOWLEDGEMENTS

The authors thank Uri Shalit, Fredrik Johannson, Adam Yala, and the anonymous reviewers for their comments that have improved the manuscript. RGK, AK, and DS are supported by a grant from SAP Corporation.

[2]Their model does not enforce the conditional independence statement $x_t \perp\!\!\!\perp \mathcal{Q}|z_t$

References

Bar-Hillel, Aharon, Hertz, Tomer, Shental, Noam, & Weinshall, Daphna. 2005. Learning a mahalanobis metric from equivalence constraints. *JMLR*.

Bauer, Matthias, Rojas-Carulla, Mateo, Bartłomiej Świątkowski, Jakub, Schölkopf, Bernhard, & Turner, Richard E. 2017. Discriminative k-shot learning using probabilistic models. *arXiv preprint arXiv:1706.00326*.

Chollet, François, *et al.* . 2015. *Keras*. `https://github.com/keras-team/keras`.

Church, Kenneth Ward, & Hanks, Patrick. 1990. Word association norms, mutual information, and lexicography. *Computational linguistics*.

Edwards, Harrison, & Storkey, Amos. 2017. Towards a neural statistician. *In: ICLR*.

Engel, Jesse, Hoffman, Matthew, & Roberts, Adam. 2018. Latent Constraints: Learning to Generate Conditionally from Unconditional Generative Models. *In: ICLR*.

Fano, Robert M. 1949. *The transmission of information*.

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

Ghahramani, Zoubin, & Heller, Katherine A. 2005. Bayesian sets. *In: NIPS*.

He, Kaiming, Zhang, Xiangyu, Ren, Shaoqing, & Sun, Jian. 2016. Deep residual learning for image recognition. *In: CVPR*.

Hinton, Geoffrey E, Dayan, Peter, Frey, Brendan J, & Neal, Radford M. 1995. The" wake-sleep" algorithm for unsupervised neural networks. *Science*.

Jeffreys, Harold. 1998. *The Theory of Probability*. OUP Oxford.

Johnson, Matthew, Duvenaud, David K, Wiltschko, Alex, Adams, Ryan P, & Datta, Sandeep R. 2016. Composing graphical models with neural networks for structured representations and fast inference. *In: NIPS*.

Kingma, Diederik, & Ba, Jimmy. 2015. Adam: A method for stochastic optimization. *In: ICLR*.

Kingma, Diederik P, & Welling, Max. 2014. Auto-encoding variational bayes. *In: ICLR*.

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

LeCun, Yann, Cortes, Corinna, & Burges, Christopher JC. 1998. *The MNIST database of handwritten digits*.

Li, Chongxuan, Zhu, Jun, Shi, Tianlin, & Zhang, Bo. 2015. Max-margin deep generative models. *Pages 1837–1845 of: Advances in neural information processing systems*.

Mishra, Nikhil, Rohaninejad, Mostafa, Chen, Xi, & Abbeel, Pieter. 2018. Meta-learning with temporal convolutions. *ICLR*.

Munkhdalai, Tsendsuren, & Yu, Hong. 2017. Meta networks. *ICML*.

Ravi, Sachin, & Larochelle, Hugo. 2016. Optimization as a model for few-shot learning. *In: ICLR*.

Rezende, Danilo Jimenez, & Mohamed, Shakir. 2015. Variational inference with normalizing flows. *In: ICML*.

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

Snell, Jake, Swersky, Kevin, & Zemel, Richard. 2017. Prototypical networks for few-shot learning. *In: NIPS*.

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

Yu, Chun-Nam John, & Joachims, Thorsten. 2009. Learning structural svms with latent variables. *In: ICML*. ACM.

Zaheer, Manzil, Kottur, Satwik, Ravanbakhsh, Siamak, Poczos, Barnabas, Salakhutdinov, Ruslan, & Smola, Alexander. 2017. Deep Sets. *arXiv preprint arXiv:1703.06114*.

Zhu, Jun, & Xing, Eric P. 2009. Maximum entropy discrimination Markov networks. *Journal of Machine Learning Research*, **10**(Nov), 2531–2569.