Learning Exponential Families

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recently much attention has been paid to implicit probability models – models defined by mapping a simple random variable through a complex transformation, often a deep neural network. These models have been used to great success for variational inference, generation of complex data types, and more. In most all of these settings, the goal has been to find a *particular member* of that model family: optimized parameters index a distribution that is close (via a divergence or classification metric) to a target distribution (such as a posterior or data distribution). Much less attention, however, has been paid to the problem of *learning a model itself*. Here we define implicit probability models with specific deep network architecture and optimization procedures in order to learn intractable exponential family models (*not* a single distribution from those models). These exponential families, which are central to some of the most fundamental problems in probabilistic inference, are learned accurately, allowing operations like posterior inference to be executed directly and generically by an input choice of natural parameters, rather than performing inference via optimization for each particular realization of a distribution within that model. We demonstrate this ability across a number of non-conjugate exponential families that appear often in the machine learning literature.

1 Introduction

Probability models, the fundamental object of Bayesian machine learning, have long challenged researchers with the tradeoff between tractability and expressivity. Though well understood that a model should be chosen to instantiate a set of assumptions and capture existing domain knowledge [1, 2, 3], for many years too-simple models were chosen for their practical advantanges (such as conditional conjugacy), which left much to be desired in terms of expressive performance and scalability of these models.

More recently the pendulum has swung, via a resurgence in models which map a latent random variable $w \sim q_0$ through a member of a highly expressive function family $\mathcal{G} = \{g_\theta : \theta \in \Theta\}$, the composition resulting in an *implicit probability model* $\mathcal{M} = \{q(g_\theta \circ w) : \theta \in \Theta\}$ (where $q(\cdot)$ is the pushforward density, i.e. the density induced on the image of the random variable w under the function g_θ). Choosing $\mathcal{G}$ to be a parameter-indexed family of neural networks has both a rich history [4, 5], and has recently been used to produce exciting results for density estimation [6, 7, 8], generation of complex data [9], variational inference [10, 11, 12], and more. A noted advantage of these implicit density network models is that in many cases they make minimal assumptions about the data generative (or posterior inference) process. On the other hand, since these models have been chosen to be generic and flexible, they can lack the classic stipulation that a model instantiates existing domain knowledge. The downsides of a too-flexible model with finite data (albeit large) – and the corresponding bias-variance benefit of a restricted model – are textbook knowledge [13,

§7.3], and work on generalization and compressibility in deep networks suggests that this broad class of function families are indeed quite large, perhaps problematically so [14].

Is all the flexibility of an implicit density network model $\mathcal{M}$ always necessary? Consider the case of variational inference, where a generative model $p(z)p_\beta(X|z)$ (latent z, observed data X) is stipulated in the classic sense to embody modeling assumptions (hierarchical model, topic model, Bayesian logistic regression, etc.). When such a model is intractable, it is increasingly common to deploy an implicit "recognition network" model for variational inference [10], which finds a $q_{\theta^*}(z) \in \mathcal{M}$ such that an evidence bound is optimized with respect to the true posterior $p(z|X)$. However, note the widely recognized fact [15] that many such true posteriors $p(z|X)$ belong to models that can be written as exponential families (albeit intractable, due to the choice of sufficient statistics $t(z)$). Some effort has been made to learn single members of exponential families from mean parameters [16], but we are focused on the natural parameterization and the model itself.

Should we be able to learn a tractable approximation to this exponential family model, we would in the very least get the bias-variance benefits of an intelligently restricted model space, and at best would get inference "for free" in the sense that we could evaluate approximate posteriors directly without separate optimization for each dataset encountered (a novel form of *amortized inference* [17, 10, 11, 18]). In this paper we aim to learn a restricted model $\mathcal{Q} = \{q(z;\eta : \eta \in H)\}$ that will be a strict subset of $\mathcal{M}$ and will closely approximate a target exponential family $\mathcal{P}$. Note the critical difference between this aim and much of the literature that seeks to learn a density $q_\theta^* \in \mathcal{M}$ (we explore this distinction in depth both algorithmically and empirically).

To proceed, we must first specify a set of models $\mathbb{Q} = \{\mathcal{Q}_\phi : \phi \in \Phi\}$, from which we can learn a single model $\mathcal{Q}_{\phi^*}$, and we must second define a sensible parameter space H of each model. To the first, we restrict Θ, the parameter space of $\mathcal{M}$, to be itself the image of a second deep *parameter network* family $\mathcal{F} = \{f_\phi : \phi \in \Phi\}$, such that $\{f_\phi(\eta) : \eta \in H\} \subset \Theta$. The second part is answered immediately by our choice of target $\mathcal{P}$, an exponential family which by definition has *natural* parameterization $\eta \in H$. Thus, appealingly, we know that H is precisely the correct parameter space for $\mathcal{Q}$ (as it defines $\mathcal{P}$), and that the image of H under f_ϕ will be of the correct dimensionality within the codomain Θ; approximation error between $\mathcal{Q}$ and $\mathcal{P}$ will be caused by the flexibility and learnability of the parameter network f_ϕ and the density network $g_{f_\phi(\eta)}$.

We define this two-network architecture, which we term an *exponential family network* (EFN), and we specify a stochastic optimization procedure over a variant of the typical Kullback-Leibler divergence. We then demonstrate the ability of EFNs to approximately learn exponential families, both known tractable families and well-used intractable families, including hierarchical Dirichlet and truncated normal Poisson families. Finally we demonstrate the utility of this approach in an example inferring the posterior distribution of the latent intensity of a point-process, given neural spike train data.

2 Exponential family networks

To define exponential family networks (EFN), we begin with relevant context for our modeling choice of exponential families (§2.1). We then describe the primary network architectural constraint and the background we leverage to satisfy that constraint (§2.2). We then introduce EFN in detail, including the optimization algorithm used for learning (§2.3). The similarities with variational inference are then explored in depth in §2.4.

2.1 Exponential families as target model $\mathcal{P}$

We will focus on a fundamental problem setup in probabilistic inference, that of a latent variable $z \in \mathcal{Z}$ with prior belief $p_0(z)$, and where we observe a dataset $X = \{x_1, ..., x_N\} \subset \mathcal{X}$ as conditionally independent draws given z. Updating our belief with data produces the posterior $p(z|X) \propto p_0(z) \prod_{i=1}^N p(x_i|z)$. This setup is shown as a graphical model in Figure 1A and produces an intractable $p(z|X)$ in all but the rare cases of known conjugacy or careful historical work (often an inversion, transformation-rejection, or similar custom numerical strategy) that has made these distributions computationally indistinguishable from tractable [19].

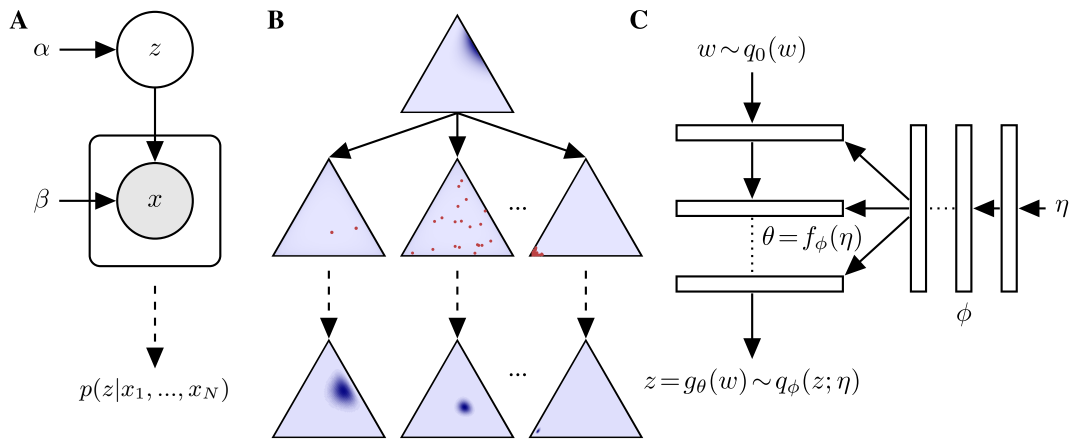

Figure 1: (A) Graphical model for conditionally iid sampling from an exponential family likelihood. (B) Hierarchical Dirichlets – prior $p_0(z)$ (top), three sample conditional Dirichlet datasets X of $N=2, N=20, N=100$ (middle), and three corresponding posteriors that themselves form an exponential family $\mathcal{P}$ (bottom). (C) Architecture for exponential family network (EFN) – density network running top to bottom; parameter network running right to left.

If we restrict our attention to priors and likelihoods that belong to exponential families $\mathcal{P} = \left\{\frac{h(\cdot)}{A(\eta)}\exp\left\{\eta^\top t(\cdot)\right\} : \eta \in H\right\}$, the posterior can be also viewed as an exponential family, albeit intractable [15]. For simplicity we will hereafter suppress the base measure $h(\cdot)$. Consider:

$$p_0(z) = \frac{1}{A_0(\alpha)}\exp\left\{\alpha^\top t_0(z)\right\} \quad , \quad p(x_i|z) = \frac{1}{A(z)}\exp\left\{\nu(z)^\top t(x_i)\right\},$$

where $t(\cdot)$ is the sufficient statistic vector, and $\nu(z)$ is the natural parameter of the likelihood in natural form [20]. The posterior then has the form:

$$p(z|x_1,...,x_N) \propto \exp\left\{\begin{bmatrix}\alpha \\ \sum_i t(x_i) \\ -N\end{bmatrix}^\top \begin{bmatrix} t_0(z) \\ \nu(z) \\ \log A(z)\end{bmatrix}\right\}, \tag{1}$$

which again is an exponential family, albeit intractable.

To give a concrete example, consider the hierarchical Dirichlet – a Dirichlet prior $z \sim Dir(\alpha)$ (of dimension $|\mathcal{Z}|$) with conditionally iid Dirichlet draws $x_i|z \sim Dir(\beta z)$, which has been considered historically [21], and is perhaps most notable for its nonparametric extension [22] (and has relevance for multi-corpus extensions of topic models [23, 24]). Figure 1B shows the prior for a given α (top), and three examples of datasets that could arise via this generative model (middle). A set of basic manipulations shows the hierarchical Dirichlet posterior $p(z|X)$ to be itself an exponential family with natural parameter $\eta = [\alpha - 1, \sum_i \log(x_i), -N]^\top$ and sufficient statistic $t(z) = [\log(z), \beta z, \log(B(\beta z))]^\top$.[1] The corresponding posteriors are shown in Figure 1B (bottom).

Note importantly that, because the likelihood was chosen to be an exponential family (which is closed under sampling), this form will not change for any choice of $|Z|$-dimensional hiearchical Dirichlet – any draw from the prior, any N, or any particular realization of observed data X (technically the prior need not be exponential family, but we leave it as such for simplicity). The exponential family is clearly sufficient for this property, and the Pitman-Koopman Lemma further clarifies that it is also necessary (under reasonable conditions) [20, §3.3.3].

[1]To be clear this model is an exponential family if β is fixed or treated as a latent variable; this fact however will not be important for the development of this paper.

The critical observation here is that, if we can approximately learn an intractable exponential family (the model itself), then it becomes trivial to perform posterior inference: we simply use the dataset to index into the natural parameter η of the intractable family, and the posterior distribution is produced.

2.2 Density networks as generic approximating family $\mathcal{M}$

Implicit probability models, which we will use for our approximating model family $\mathcal{M}$, can be defined by any base random variable $w \sim p_0$ mapped through any measurable, parameter-indexed function family $\mathcal{G} = \{g_\theta : \theta \in \Theta\}$; we denote the induced density on $z = g_\theta(w)$ as $q_\theta(z)$. Though trivial to sample from $q_\theta(z)$ for any choice of family $\mathcal{G}$, we here additionally require that we be able to explicitly calculate $q_\theta(z)$. This goal can be readily achieved by designing $\mathcal{G}$ to contain only bijective functions, ideally with a Jacobian form that is convenient to compute. Designing that bijective $\mathcal{G}$ as a deep neural network family, as we do here, is a well-established idea that has recently seen many variants and applications [5, 25, 26, 7, 6, 27, 28, 8, 29]. Specifically, let $z = g_\theta(w) = g_L \circ ... \circ g_1(w)$ for bijective vector-valued functions g_ℓ (surpressing θ), and denote $J_\theta^\ell(z)$ as the Jacobian of the function g_ℓ at the layer activation corresponding to z. Then we have:

$$q_\theta(z) = q_0\left(g_1^{-1} \circ ... \circ g_L^{-1}(z)\right) \prod_{\ell=1}^{L} \frac{1}{|J_\theta^\ell(z)|}.$$

The specific form of the layers g_ℓ can be chosen based on empirical considerations; we clarify our choice in §3. For the remainder (and to avoid confusion when we introduce a second network) we call this deep bijective neural architecture the *density network*; this network is shown vertically oriented (flowing from w down to z) in Figure 1C.

This density network induces the model $\mathcal{M} = \{q(g_\theta \circ w) : \theta \in \Theta\}$, which previous work has searched to find a single optimized distribution (such as a posterior or data generative density), on the assumption and subsequent empirical evidence that the target exponential family member is close to (or approximately belongs to) $\mathcal{M}$. We make the same assumption for the exponential family itself and seek to intelligently restrict $\mathcal{M}$ in order to learn the exponential family.

2.3 Exponential family networks as approximating model $\mathcal{Q}$

Having introduced our target model $\mathcal{P}$, an exponential family with natural parameters $\eta \in H$, and the density network family $\mathcal{M}$, we now seek to learn $\mathcal{Q} \approx \mathcal{P}$, where $\mathcal{Q} \subset \mathcal{M}$. To do so we will parameterize θ, the parameters of the density network, as the image of a second *parameter network* family $\mathcal{F} = \{f_\phi : H \to \Theta, \phi \in \Phi\}$. This network is shown flowing from right to left in Figure 1C. Using a second meta-network to aid or restrict network learning has been used in a variety of settings; a few examples include parameterizing the optimization algorithm in the so-called "learning to learn" setting [30], and a more closely related work that used a second network to condition on observations for local latent variational inference [27], a connection which we explore closely in the following section.

Any choice of parameter network parameters ϕ induces a $|H|$-dimensional submanifold (the image $f_\phi(H)$) of the density network parameter space Θ, and as such defines a restricted model $\mathcal{Q}_\phi = \left\{q_{f_\phi}(z;\eta) : \eta \in H\right\} \subset \mathcal{M}$; by our choice of H as the natural parameter space of the exponential family target $\mathcal{P}$, this model restriction is at least of the correct dimensionality. Our goal then is to search over the implied set of models $\mathbb{Q} = \{\mathcal{Q}_\phi : \phi \in \Phi\}$ to find an optimal ϕ^* such that $\mathcal{Q}_{\phi^*} \approx \mathcal{P}$.

Given the connections between the exponential family and Shannon entropy, we will measure the error between $\mathcal{Q}_\phi$ and $\mathcal{P}$ with Kullback-Leibler divergence. Consider for the moment a fixed choice of natural parameter η; we seek to minimize, over ϕ:

$$D\left(q_\phi(z;\eta)||p(z;\eta)\right) \propto \mathbb{E}_{q_\phi}\left(\log q_\phi(z;\eta) - \eta^\top t(z)\right) = \mathbb{E}_{q_\phi}\left(q_0\left(g_\theta^{-1}(z)\right) + \sum_{\ell=1}^{L} \log |J_\theta^\ell(z)| - \eta^\top t(z)\right),$$

where again we note that $\theta = f_\phi(\eta)$, and thus for a fixed eta, this objective depends only on ϕ. Indeed, the target $\eta^\top t(z)$ is linear in η (an obvious restatement of the log-linear exponential family form), giving us some hope that we may be able to learn this model. As a side note, this objective can also produce approximations of the log partition (as the intercept term implied by this linear target), which we have found to be reasonably accurate, though nuanced schemes are likely appropriate [31]; we do not explore that further here.

Of course we seek to approximate not just a single target exponential family member ($p(z;\eta)$ for a fixed η), but rather the entire model $\mathcal{P} = \{p(z;\eta) : \eta \in H\}$. For optimization we thus need to introduce a distribution $p(\eta)$ (for sampling), leading to the objective:

$$\underset{\phi}{\operatorname{argmin}}\, \mathbb{E}_{p(\eta)} \left(D\left(q_\phi(z;\eta) || p(z;\eta) \right) \right) = \underset{\phi}{\operatorname{argmin}}\, D\left(q_\phi(z;\eta)p(\eta) || p(z;\eta)p(\eta) \right).$$

Unbiased estimates of this objective are immediate: $q_\phi(z;\eta)$ is sampled by computing calculating the density network parameters $\theta = f_\phi(\eta)$ (using the parameter network), sampling the latent $w \sim p_0(w)$, and running that w through the density network; $p(\eta)$ is user defined and thus trivial to sample. Stochastic optimization can then be carried out on the estimator:

$$\mathbb{L}(\phi) = \frac{1}{K}\frac{1}{M}\sum_{k=1}^{K}\sum_{m=1}^{M}\left(q_0\left(g_{\theta^k}^{-1}\left(z^m\right)\right) + \sum_{\ell=1}^{L}\log|J_{\theta^k}^{\ell}\left(z^m\right)| - \eta_k^\top t\left(z^m\right)\right), \tag{2}$$

where $\theta^k = f_\phi\left(\eta_k\right)$. Successful optimization over ϕ should thus result in $\mathcal{Q}_{\phi^*} \in \mathbb{Q}$ that accurately approximates the target exponential family; that is, $\mathcal{Q} \approx \mathcal{P}$. We call this two-network architecture and optimization an exponential family network (EFN). What remains for empirical implementation is to make particular choices of hyperparameters, network layers, and optimization algorithm, which we specify in §3 below.

2.4 Relation to variational inference

A tremendous amount of work in recent years has gone into variational inference (VI), and its similarity to EFN warrants careful attention. In the following, we aim to carefully (and somewhat pedantically) dissect this question. As such, though EFN can address any target exponential familiy, to bring us closest to VI let us here restrict the EFN target model $\mathcal{P}$ to be a family of posterior distributions.

The typical role of variational inference is to infer an approximate posterior $q_\phi(z) \approx p(z|X)$. In this setting, the difference with EFN is stark, in so much as VI learns this single posterior approximation, whereas the main goal of the EFN is to approximate the model $\mathcal{P} = p_\eta(z|X) : \eta in H$: to learn the family of distributions. More recently, much focus has gone into the particular instance of VI for local variables z_i, for example $\prod_{i=1}^{N} p(z_i)p(x_i|z_i)$ (such as a variational autoencoder [10]) or $p(u)\prod_{i=1}^{N} p(z_i|u)p(x_i|z_i)$ (latent Dirichlet allocation being a canonical example [23, 32]), the result of which is often an amortized inference/recognition network that produces a local variational distribution $q_{\phi^*}(z_i|x_i)$. This local variational distribution is typically parameterized explicitly: the inference network $\mu_\phi(x_i)$ induces a local parametric distribution, often a Gaussian $q(z_i|x_i) \sim \mathcal{N}\left(z_i;\mu_\phi(x_i)\right)$ [10, for example]. Viewed this way, local-latent-variable VI methods induce a model $\{q_{\phi^*}(z_i|x_i) : x_i \in X\}$ for a finite dataset X. In that sense, EFN and VI are similar 'model learning' approaches. Even more closely, as part of a long-standing desire to add structure to VI beyond mean-field (classically [33, 34]; more recently [35, 36], to name but a few), in several cases a inference network has been used to parameterize a deep implicit model (in a two-network inference architecture, to say nothing of whether or not the generative model itself is a deep implicit model); closest to the EFN architecture is [27] (cf. Figure 2 of [27] with Figure 1C here). Thus EFN (when used for posterior families) can be seen as a close generalization of VI.

However, even accepting this VI-as-a-model view, the difference between the finite dataset X and the natural parameter space H persists when viewed at a mechanical level; well-known are the overfitting/generalization issues associated with a finite dataset compared with access to a distribution $p(\eta)$. Thus one goal of EFN is to allow the model $\mathcal{Q}_{\phi^*} \approx \mathcal{P}$ to be learned in the absence of a finite dataset, such that inference on that dataset can then be executed without concerns of overfitting to that set (and of course without having to run a VI optimization for every new dataset). Perhaps more importantly, the "model" implied by VI is parameterized by x_i, and indeed the inference network takes x_i as input. The EFN on the other hand is considerably more general: as Equation 1 shows, the posterior includes the natural parameters of the prior, allowing the EFN architecture to learn across a more general setting that VI can not (since any VI inference network is only parameterized by data). One final difference made clear by Equation 1 is that the observations are given to the EFN *in natural form* (that is, $t(x_i)$, not x_i) [20]. This choice is a novel insight: by exploiting the known sufficiency of $t(x_i)$ in the target model $\mathcal{P}$, some difference in performance for VI may be observed. We explore this empirically in the following section.

Accordingly, while EFN and VI do at a high level bear multiple similarities, the differences are both material and provoke interesting speculation about means to improve both VI and EFN.

3 Results

We perform a number of experiments to investigate the performance of EFN. First, we test the ability of EFN to approximate the target model $\mathcal{P}$ when this model is a known, tractable exponential family: this choice provides a simple ground truth and calibrates us to expected performance vs alternatives. The main advantage of learning an EFN is to make tractable a previously intractable exponential family (at least approximately). This confers major benefits in terms of test-time: for example, rather than optimization needing to be run for variational inference with each particular dataset realized from a model class, EFN will allow immediate lookup. This benefit is orders of magnitude and is not instructive to view, so here we focus our analyses on the costs of doing so: what approximation loss is suffered when learning a whole family vs a single distribution.

To make this comparison, we use two alternatives. First, we restrict our algorithm to a single η; that is, $K = 1$ in Equation 2, and further that choice of η is fixed throughout the course of optimization (not stochastically sampled at every time). This is then a direct comparison that asks, given the same exact implicit model architecture, what cost is paid to learn a full model vs a single distribution. We call this alternative EFN1, which optimizes over ϕ as in the EFN. Second, it seems unnecessary to carry around an entire parameter network $f_\phi(\eta)$ if that η will not change; thus our second alternative (which is in some ways mechanically closest to traditional VI) is to dispose of the parameter network and train the density network directly over θ (again with a deterministic choice of a single η); we call this alternative NF1.

We also must make some particular architectural choices for these experiments. We considered a variety of density network architectures; in all the results we use the planar flow layer introduced in [27]. The parameter network was given $\tanh$ nonlinearities. In many of the results below we will analyze EFNs across a range of problem dimensionality D (that is, $z \in \mathcal{Z} \subseteq \mathbb{R}^D$). In all cases then we have also D planar flow layers in the density network, with $2D + 1$ density network parameters per layer. In analyses where D was less than 20, 20 planar flows were used. The number of layers in the parameter network scaled as the square root of D, with a minimum of 4 layers, and the number of units per layer scaled linearly from the input to the number of density network parameters. Models were trained using the ADAM optimizer algorithm, with learning rates ranging from 10^{-3} to 10^{-5} and from 20,000 to 50,000 iterations. These choices were made so that model performance saturated, and were held constant within comparative analyses. All code was implemented in tensorflow, and will be available at www.github.com/<anonymous>.

3.1 Tractable exponential families

Here we study the Dirichlet, Gaussian, and inverse-Wishart families, which offer a known ground truth and intuition about the range of performance that EFN – learning a model – can see with respect to its single-distribution counterparts (NF1 and EFN1).

First, to validate the basic EFN approach, we train the $D = 25$-dimensional Dirichlet family. We chose $p(\eta)$, the prior on the α parameter vector of the Dirichlet, as $\alpha_i \sim U\,[.5, 5.0]$. The number of η samples K at each iteration was 100, and the minibatch size in z was $M = 1000$. Figure 2 shows a high accuracy fit to this Dirichlet model: Figures 2A and 2B shows rapid convergence to high r^2 and low Kullback-Leibler divergence. r^2 is a convenient metric in so much as we are here doing distribution regression, so we calculate the coefficient of determination between the model predictions $q_\phi(z_i; \eta_k)$ and their known targets $\eta_k^\top t(z_i)$. We can then perform a standard MMD-based kernel two-sample test [37] between distributions chosen from $\mathcal{P}$ and $\mathcal{Q}_{\phi^*}$: the unstructured distribution of p values clarifies that the EFN model $\mathcal{Q}_{\phi^*}$ is not statistically significantly different than the true target Dirichlet family $\mathcal{P}$ (using a test with 50 samples).

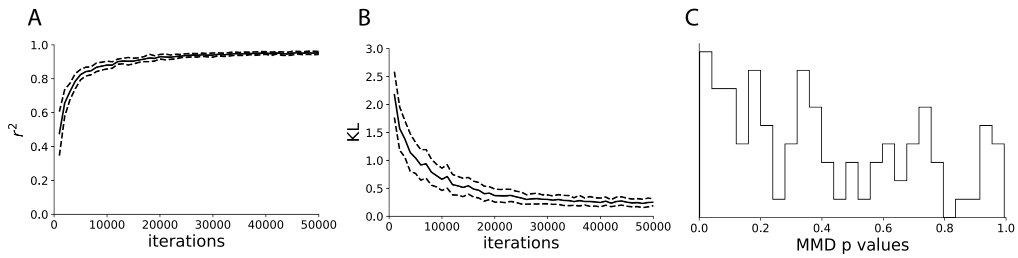

Figure 2: 25-dimensional Dirichlet exopnential family network. (A) Distribution of r^2 between log density of EFN samples and ground truth across choices of η throughout optimization. (B) Distribution of KL divergence throughout optimization. (C) Distribution of maximum mean discrepancy p-values between EFN samples and ground truth after optimization.

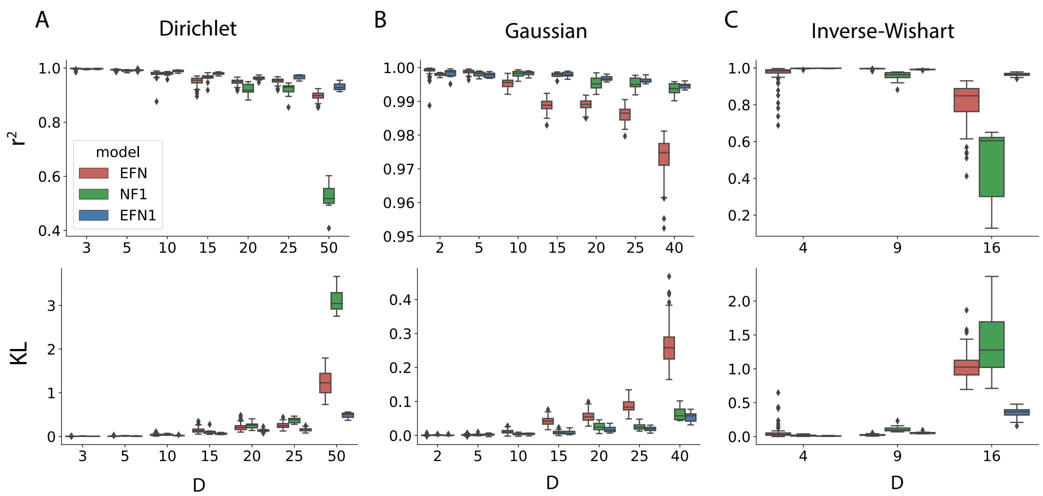

Figure 3: Scaling exponential family networks: D denotes the dimensionality of the family being learned, and comparisons are between EFN and its $K = 1$ alternatives NF1 and EFN1 (see text). (A) Dirichlet family (B) Gaussian family (C) Inverse-Wishart family.

Second, in Figure 3 we consider how this performance scales across dimensionality. Consider EFN vs EFN1, where again the only difference is that EFN attempts to learn the entire model (as in $\eta \in H$), whereas EFN1 chooses a single η and thus learns a single distribution. In both the Dirichlet and the Gaussian (Figure 3A and 3B), there is very minor (but statistically significant) loss from the EFN1 to EFN (but note the zoomed axis in Figure 3B; this difference is less than it may appear). This is quite encouraging: though training an entire model as opposed to a single distribution, performance holds up adequately. If this performance level is adequate, using such a model is immediate; of course, failing that, the EFN could be used on a case by case basis to initialize the parameters $\theta_0 = f_\phi(\eta)$ for further optimization in θ. Performance in the inverse-Wishart is considerably less impressive when comparing the EFN to the EFN1, though we have found no satisfactory explanation for the shortcoming. It is also important to note that the distribution $p(\eta)$ can have material consequence on performance: the less entropic that distribution, the closer EFN gets to EFN1 by definition. The Dirichlet family has in our experience been robust to that choice, though perhaps surprisingly the Gaussian family has been less so (we swept the degrees of freedom of a Wishart prior on the covariance of the Gaussian $\nu = 5D, 100D, 1000D$; the middle choice is shown here, the other two having very strong and very poor performance). Quite surprising is the performance of NF1. As a reminder the NF1 trains the density network directly over θ. One would think that, in so much as θ is typically of lower dimension than ϕ, that the NF1 would fit more easily; this expectation was only found in Figure 3B, though in Figure 3A and 3C EFN1 and EFN tended to outperform and scale better than NF1.

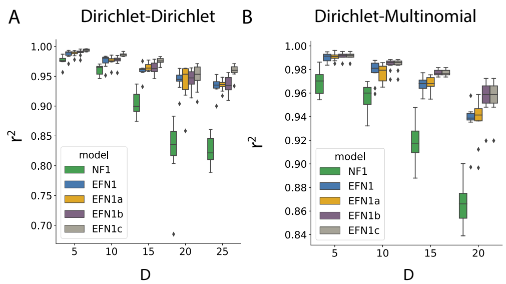

Figure 4: Dirichlet families. See text.

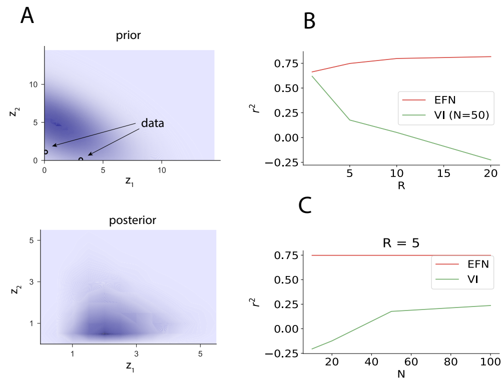

Figure 5: Truncated normal Poisson and neural spike train analysis. See text.

3.2 The hierarchical Dirichlet family

Of course the main interest of an EFN is to learn intractable exponential families. We here consider the hierchical Dirichlet family (as introduced in §2.1 and Figure 1A,B) to explore empirically the detailed connections of EFN to variational inference.

3.3 The truncated normal Poisson family, and neural spike train analysis

The normal family is the ubiquitous prior for real valued parameters, but it does not match well with the nonnegativity requirements of the intensity measure required of certain distributions, most notably the Poisson. Truncated normal and log Gaussian Cox Processes have been used numerous times in machine learning, and all have required attention to approximate inference in this fundamentally nonconjugate model; furthermore, very many of these examples have been used to analyze the latent firing intensity of neural spike train data [38, 39, 40, 41].

4 Conclusion

We have approached the problem of learning an exponential family, rather than a particular member of that family. We did so using a deep density network as an implicit probability model, the parameters of which are the image of the natural parameters of the target exponential family under another deep neural network. We demonstrated high quality empirical performance across a range of dimensionalities, making a number of previously intractable distributions, including posterior distributions, *approximately tractable*. We have scrutinized the connections between our exponential family networks and variational inference, producing surprising and at times puzzling results that are worthy of meaningful follow up study. In all, we have demonstrated the ability to capture performance gains and massive test-time advantage by sensibly restricting the space of an implicit probability model.

References

[1] Andrew Gelman, John B Carlin, Hal S Stern, David B Dunson, Aki Vehtari, and Donald B Rubin. *Bayesian data analysis*, volume 2. CRC press Boca Raton, FL, 2014.

[2] Joshua B Tenenbaum, Thomas L Griffiths, and Charles Kemp. Theory-based bayesian models of inductive learning and reasoning. *Trends in cognitive sciences*, 10(7):309–318, 2006.

[3] Peter McCullagh. What is a statistical model? *The Annals of Statistics*, 30(5):1225–1267, 2002.

[4] Peter Dayan, Geoffrey E Hinton, Radford M Neal, and Richard S Zemel. The helmholtz machine. *Neural computation*, 7(5):889–904, 1995.

[5] D. J. C. MacKay and M. N. Gibbs. Density networks. In *Statistics and Neural Networks*, pages 129–146. Oxford, 1997.

[6] Benigno Uria, Iain Murray, and Hugo Larochelle. Rnade: The real-valued neural autoregressive density-estimator. In *Advances in Neural Information Processing Systems*, pages 2175–2183, 2013.

[7] Oren Rippel and Ryan Prescott Adams. High-dimensional probability estimation with deep density models. *arXiv preprint arXiv:1302.5125*, 2013.

[8] George Papamakarios, Iain Murray, and Theo Pavlakou. Masked autoregressive flow for density estimation. In *Advances in Neural Information Processing Systems*, pages 2335–2344, 2017.

[9] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014.

[10] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv*, 12 2013.

[11] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

[12] Michalis Titsias and Miguel Lázaro-Gredilla. Doubly stochastic variational bayes for non-conjugate inference. In *International Conference on Machine Learning*, pages 1971–1979, 2014.

[13] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[14] Wenda Zhou, Victor Veitch, Morgane Austern, Ryan P Adams, and Peter Orbanz. Compressibility and generalization in large-scale deep learning. *arXiv preprint arXiv:1804.05862*, 2018.

[15] Martin J Wainwright, Michael I Jordan, et al. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1–2):1–305, 2008.

[16] Gabriel Loaiza-Ganem, Yuanjun Gao, and John P Cunningham. Maximum entropy flow networks. *International Conference on Learning Representations*, 2017.

[17] Samuel Gershman and Noah Goodman. Amortized inference in probabilistic reasoning. In *Proceedings of the Annual Meeting of the Cognitive Science Society*, volume 36, 2014.

[18] Andreas Stuhlmüller, Jacob Taylor, and Noah Goodman. Learning stochastic inverses. In *Advances in neural information processing systems*, pages 3048–3056, 2013.

[19] Luc Devroye. *Non-uniform random variate generation*. Springer-Verlag, New York, 1986.

[20] Christian Robert. *The Bayesian choice: from decision-theoretic foundations to computational implementation*. Springer Science & Business Media, 2007.

[21] David JC MacKay and Linda C Bauman Peto. A hierarchical dirichlet language model. *Natural language engineering*, 1(3):289–308, 1995.

[22] Yee Whye Teh, Michael I Jordan, Matthew J Beal, and David M Blei. Hierarchical dirichlet processes. *Journal of the American Statistical Association*, 101(476):1566–1581, 2006.

[23] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022, 2003.

[24] Jonathan K Pritchard, Matthew Stephens, and Peter Donnelly. Inference of population structure using multilocus genotype data. *Genetics*, 155(2):945–959, 2000.

[25] Leemon Baird, David Smalenberger, and Shawn Ingkiriwang. One-step neural network inversion with pdf learning and emulation. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 2, pages 966–971. IEEE, 2005.

[26] Esteban G Tabak, Eric Vanden-Eijnden, et al. Density estimation by dual ascent of the log-likelihood. *Communications in Mathematical Sciences*, 8(1):217–233, 2010.

[27] Danilo Jimenez Rezende and Shakir Mohamed. Variational inference with normalizing flows. *arXiv preprint arXiv:1505.05770*, 2015.

[28] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[29] Jörn-Henrik Jacobsen, Arnold Smeulders, and Edouard Oyallon. i-revnet: Deep invertible networks. *arXiv preprint arXiv:1802.07088*, 2018.

[30] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, and Nando de Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in Neural Information Processing Systems*, pages 3981–3989, 2016.

[31] George Papamakarios and Iain Murray. Distilling intractable generative models. In *Probabilistic Integration Workshop at Neural Information Processing Systems*, 2015.

[32] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017.

[33] Lawrence K Saul and Michael I Jordan. Exploiting tractable substructures in intractable networks. In *Advances in neural information processing systems*, pages 486–492, 1996.

[34] David Barber and Wim Wiegerinck. Tractable variational structures for approximating graphical models. In *Advances in Neural Information Processing Systems*, pages 183–189, 1999.

[35] Matthew Hoffman and David Blei. Stochastic structured variational inference. In *Artificial Intelligence and Statistics*, pages 361–369, 2015.

[36] Dustin Tran, David Blei, and Edo M Airoldi. Copula variational inference. In *Advances in Neural Information Processing Systems*, pages 3564–3572, 2015.

[37] Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(Mar):723–773, 2012.

[38] John P Cunningham, Krishna V Shenoy, and Maneesh Sahani. Fast gaussian process methods for point process intensity estimation. In *Proceedings of the 25th international conference on Machine learning*, pages 192–199. ACM, 2008.

[39] John P Cunningham, M Yu Byron, Krishna V Shenoy, and Maneesh Sahani. Inferring neural firing rates from spike trains using gaussian processes. In *Advances in neural information processing systems*, pages 329–336, 2008.

[40] Ryan Prescott Adams, Iain Murray, and David JC MacKay. Tractable nonparametric bayesian inference in poisson processes with gaussian process intensities. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 9–16. ACM, 2009.

[41] Yuanjun Gao, Evan W Archer, Liam Paninski, and John P Cunningham. Linear dynamical neural population models through nonlinear embeddings. In *Advances in Neural Information Processing Systems*, pages 163–171, 2016.