LEARNING DEEP LATENT-VARIABLE MRFS WITH AMORTIZED BETHE FREE ENERGY MINIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

While much recent work has targeted learning deep discrete latent variable models with variational inference, this setting remains challenging, and it is often necessary to make use of potentially high-variance gradient estimators in optimizing the ELBO. As an alternative, we propose to optimize a non-ELBO objective derived from the Bethe free energy approximation to an MRF's partition function. This objective gives rise to a saddle-point learning problem, which we train inference networks to approximately optimize. The derived objective requires no sampling, and can be efficiently computed for many MRFs of interest. We evaluate the proposed approach in learning high-order neural HMMs on text, and find that it often outperforms other approximate inference schemes in terms of true held-out log likelihood.

1 INTRODUCTION

There has been much recent interest in learning deep generative models with discrete latent variables (Mnih & Gregor, 2014; Mnih & Rezende, 2016; Jang et al., 2017; Maddison et al., 2017; Kaiser et al., 2018; Lee et al., 2018, *inter alia*), especially in the case where these latent variables have structure – that is, where the interdependence between the discrete latents is modeled. Most recent work has focused on learning these models with variational inference (Jordan et al., 1999), and in particular with variational autoencoders (VAEs) (Kingma & Welling, 2014; Rezende et al., 2014).

Variational inference has a number of convenient properties, including that it involves the maximization of the evidence lower-bound (ELBO), a lower bound on the log marginal likelihood of the data. At the same time, when learning models with discrete latent variables, variational inference may require the use of potentially high-variance gradient estimators, which are obtained during learning by sampling from the variational posterior.

In this paper we investigate learning discrete latent variable models with an alternative objective to the ELBO. In particular, we propose to approximate the intractable log marginal likelihood with an objective deriving from the Bethe free energy (Bethe, 1935), a quantity which is intimately related to loopy belief propagation (LBP) (Pearl, 1986; Yedidia et al., 2001; 2003), and which is the basis for "outer approximations" to the marginal polytope (Wainwright & Jordan, 2008). The Bethe free energy is attractive because if all the factors in the factor graph associated with the model have low *degree*, it can often be evaluated efficiently, without any need for approximation by sampling (see Section 2). Of course, requiring all factors in the factor graph to be of low degree severely limits the expressiveness of directed graphical models. It does not, however, limit the expressiveness of markov random fields (MRFs) (i.e., undirected graphical models) as severely, since we can simply have an extremely loopy MRF, with arbitrary pairwise factors; see Figure 1 (c) and Section 2.2.

We accordingly propose to learn deep, undirected graphical models with latent variables, using a saddle-point objective that makes use of the Bethe free energy approximation to the model's partition functions. We further amortize inference by using "inference networks" (Srikumar et al., 2012; Kingma & Welling, 2014; Johnson et al., 2016; Tu & Gimpel, 2018) in optimizing the saddle point objective. Unlike the ELBO, our objective will not form a lower bound on the log marginal likelihood, but an approximation to it. At the same time (and unlike other recent work on MRFs with a variational flavor (Kuleshov & Ermon, 2017; Li et al., 2019)), this objective can be optimized efficiently, without sampling, and in our experiments it outperforms other approximate inference methods in terms of held out log likelihood. Our experiments also indicate, however, that current approximate inference methods significantly underperform learning with exact inference, even on relatively simple models such as HMMs.

2 Background and Notation

2.1 Markov Random Fields

Let $\mathcal{G} = (\mathcal{V} \cup \mathcal{F}, \mathcal{E})$ be a factor graph (Frey et al., 1997; Kschischang et al., 2001), with $\mathcal{V}$ the set of variable nodes, $\mathcal{F}$ the set of factor nodes, and $\mathcal{E}$ the set of undirected edges between elements of $\mathcal{V}$ and elements of $\mathcal{F}$; see Figure 1 (b) and (c) for two examples. We will refer collectively to variables in $\mathcal{V}$ that are always observed as $\mathbf{x}$, and to variables which are never observed as $\mathbf{z}$. In an MRF, the joint distribution then factorizes as

$$P(\mathbf{x}, \mathbf{z};\, \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{\alpha} \Psi_\alpha(\boldsymbol{x}_\alpha, \boldsymbol{z}_\alpha;\, \boldsymbol{\theta}),$$

where α indexes elements of $\mathcal{F}$, the potentials functions Ψ_α are assumed to always be positive and are parameterized by $\boldsymbol{\theta}$, and $Z(\boldsymbol{\theta})$ is the partition function, given by:

$$Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}'} \sum_{\boldsymbol{z}'} \prod_{\alpha} \Psi_\alpha(\boldsymbol{x}'_\alpha, \boldsymbol{z}'_\alpha;\, \boldsymbol{\theta}).$$

We use the notation $\mathbf{x}_\alpha$ and $\mathbf{z}_\alpha$ to denote the subvectors of $\mathbf{x}$ and $\mathbf{z}$, respectively, that participate in the factor indexed by α. (We will similarly denote subvectors of the realizations of $\mathbf{x}$ and $\mathbf{z}$ as $\boldsymbol{x}_\alpha$ and $\boldsymbol{z}_\alpha$). For example, if we had $\Psi_{22}(\boldsymbol{x}, \boldsymbol{z}) = \Psi_{22}(\boldsymbol{x}_{22}, \boldsymbol{z}_{22}) = \exp(x_1 \times x_3 + z_4)$, then $\boldsymbol{x}_{22} = [x_1; x_3]$ and $\boldsymbol{z}_{22} = z_4$.[1]

Marginalizing out the unobserved variables yields:

$$P(\mathbf{x};\, \boldsymbol{\theta}) = \sum_{\boldsymbol{z}'} P(\boldsymbol{x}, \boldsymbol{z}';\, \boldsymbol{\theta}) = \sum_{\boldsymbol{z}'} \frac{1}{Z(\boldsymbol{\theta})} \prod_{\alpha} \Psi_\alpha(\boldsymbol{x}_\alpha, \boldsymbol{z}'_\alpha;\, \boldsymbol{\theta}) = \frac{Z(\boldsymbol{x}, \boldsymbol{\theta})}{Z(\boldsymbol{\theta})},$$

where $Z(\boldsymbol{x}, \boldsymbol{\theta}) = \sum_{\boldsymbol{z}'} \prod_{\alpha} \Psi_\alpha(\boldsymbol{x}_\alpha, \boldsymbol{z}'_\alpha;\, \boldsymbol{\theta})$ is the partition function with $\boldsymbol{x}'$ "clamped" to $\boldsymbol{x}$ (that is, it is the partition function of $P(\boldsymbol{z} \,|\, \boldsymbol{x};\, \theta)$). When learning a latent variable UGM, it is therefore natural to minimize

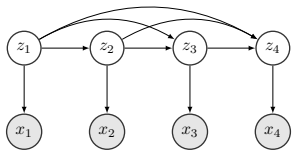
$$-\log P(\boldsymbol{x};\, \boldsymbol{\theta}) = \log Z(\boldsymbol{\theta}) - \log Z(\boldsymbol{x}, \boldsymbol{\theta}) \tag{1}$$

with respect to $\boldsymbol{\theta}$.

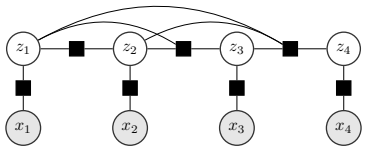
2.2 Bethe Free Energy

The Bethe free energy (Bethe, 1935) is a function of a parameterized factor graph and its marginals, which can be interpreted as an approximation of its clamped or unclamped partition function. To define it, first

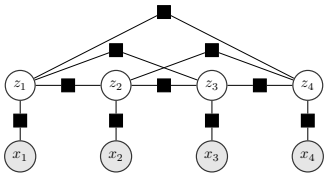
[1] To minimize notation, we will also allow subvectors to be empty.



(a) A directed 3rd order HMM.



(b) Factor graph of a directed 3rd order HMM.



(c) Factor graph of an undirected HMM with only pairwise factors, but 3rd order dependencies.

Figure 1

let $\boldsymbol{\tau}_\alpha(\boldsymbol{x}_\alpha, \boldsymbol{z}_\alpha) \in [0,1]$ be the marginal probability of random subvectors $\mathbf{x}_\alpha$ and $\mathbf{z}_\alpha$ taking on the values $\boldsymbol{x}_\alpha$ and $\boldsymbol{z}_\alpha$, respectively, which can be obtained by marginalizing out all variables that do not participate in the factor α. We will refer to the vector consisting of the concatenation of the marginal probabilities for each instantiation of each factor as $\boldsymbol{\tau} \in [0,1]^n$. As a concrete example, consider the 10 factors in Figure 1 (c): if all the x variables and all the z variables can take on only two possible values, then since each factor is pairwise (i.e., considers only two variables), there are 2^2 possible settings for each factor, and thus 2^2 corresponding marginals. In total, we then have 10×4 marginals and so $n = 40$ and thus $\boldsymbol{\tau} \in [0,1]^{40}$.

Following Yedidia et al. (2003), the Bethe free energy is then defined as

$$\begin{aligned} F(\boldsymbol{\tau}) &= \mathrm{KL}[q_{\boldsymbol{\tau}}(\boldsymbol{x}, \boldsymbol{z}) || P(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta})] - \log Z(\boldsymbol{\theta}) \\ &= \sum_{\alpha} \sum_{\boldsymbol{x}'_\alpha, \boldsymbol{z}'_\alpha} \boldsymbol{\tau}_\alpha(\boldsymbol{x}'_\alpha, \boldsymbol{z}'_\alpha) \log \frac{\boldsymbol{\tau}_\alpha(\boldsymbol{x}'_\alpha, \boldsymbol{z}'_\alpha)}{\Psi_\alpha(\boldsymbol{x}'_\alpha, \boldsymbol{z}'_\alpha)} - \sum_{v \in \mathcal{V}} (\mathrm{ne}(x_v) - 1) \sum_{x'_v} \boldsymbol{\tau}(x'_v) \log \boldsymbol{\tau}(x'_v), \end{aligned} \tag{2}$$

where we use the notation $q_{\boldsymbol{\tau}}(\boldsymbol{x}, \boldsymbol{z})$ to refer to a distribution with marginals $\boldsymbol{\tau}$, and where $\mathrm{ne}(x_v)$ refers to the number of factor-neighbors node v has in the graphical model. The second line in equation 2 can be derived from the first by rewriting $q_{\boldsymbol{\tau}}(\boldsymbol{x}, \boldsymbol{z})$ as a product of its marginals divided by a (different) product of its marginals, which is always possible for distributions represented by tree-structured graphical models (Wainwright & Jordan, 2008), and then simplifying the resulting expressions; see Gormley & Eisner (2014) for an explicit derivation.

Crucially, since $\log Z(\boldsymbol{\theta})$ does not depend on $\boldsymbol{\tau}$, in the case of a tree-structured graphical model we have $\min_{\boldsymbol{\tau}} F(\boldsymbol{\tau}) = -\log Z(\boldsymbol{\theta})$, since the KL divergence vanishes when $\boldsymbol{\tau}$ matches the marginals of $P(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\theta})$. In the case where the graphical model is *not* tree-structured, we may still define $F(\boldsymbol{\tau})$ as in equation 2, but in general we will only have $\min_{\boldsymbol{\tau}} F(\boldsymbol{\tau}) \approx -\log Z(\boldsymbol{\theta})$ (see Willsky et al. (2008), Weller & Jebara (2014), and Weller et al. (2014) for more precise characterizations of the approximation), since the marginals $\boldsymbol{\tau}$ may not correspond to any distribution, and equation 2 no longer corresponds to a true KL divergence.

Nonetheless, the Bethe approximation may often work well in practice (Yedidia et al., 2003; Meshi et al., 2009). Indeed, Yedidia et al. (2001) show that loopy belief propagation corresponds to finding fixed points of the constrained optimization problem $\min_{\boldsymbol{\tau} \in \mathcal{C}} F(\boldsymbol{\tau})$, where $\mathcal{C}$ contains "pseudo-marginal" vectors $\boldsymbol{\tau}$ that meet the following constraints: (1) each marginal distribution contained in $\boldsymbol{\tau}$ consists of positive elements that sum to one, and (2) the marginals contained in $\boldsymbol{\tau}$ are locally consistent in the sense that for any random

variable v in the graph, and for any two factors α, β involving v, we have

$$\sum_{\boldsymbol{x}'_\alpha, \boldsymbol{z}'_\alpha : \mathrm{v}=k} \boldsymbol{\tau}_\alpha(\boldsymbol{x}'_\alpha, \boldsymbol{z}'_\alpha) = \sum_{\boldsymbol{x}'_\beta, \boldsymbol{z}'_\beta : \mathrm{v}=k} \boldsymbol{\tau}_\beta(\boldsymbol{x}'_\beta, \boldsymbol{z}'_\beta)$$

for all values k that v might take on. In other words, the marginal probability of any random variable v taking on a particular value k is consistent between all factors in which v participates. To use the variable z_1 in Figure 1 (c) as an example, local consistency requires that

$$\sum_{z'_2} \Psi_4(z_1 = k, z'_2) = \sum_{z'_2} \Psi_2(z_1 = k, z'_3) = \sum_{z'_4} \Psi_1(z_1 = k, z'_4) = \sum_{x'_1} \Psi_7(x'_1, z_1 = k),$$

where factors have been ordered top-down, left-to-right in the factor graph in Figure 1. Note that there are pseudo-marginals $\boldsymbol{\tau} \in \mathcal{C}$ that do not correspond to the marginals of any distribution; see Wainwright & Jordan (2008) for an example.

The attractive feature of the Bethe approximation in equation 2 for our purposes is that while it is exponential in the *degree* of each factor (because it must consider every marginal), it is only linear in the number of factors. Thus, evaluating the Bethe free energy of a factor graph with a large number of small-degree (e.g., pairwise) factors remains tractable. As noted above, while this restriction severely limits the expressiveness of *directed* graphical models, MRFs are free to have arbitrary pairwise dependence. This can be seen, for example, in Figure 1, which shows two different factor graphs for a 3rd order HMM (Rabiner, 1989). While Figure 1 (b) shows the factor graph corresponding to a directed 3rd order HMM, Figure 1 (c) shows a possible factor graph for an undirected HMM-style model, which contains only pairwise factors. Note that while Figure 1 (c) is less expressive than its directed counterpart, it still models, for instance, the dependence between z_4 and all the other variables.

3 An Unsupervised Objective

We now consider learning by making use of the Bethe approximation to minimize an approximation to the log marginal likelihood of equation 1. In particular, from the previous section we have that

$$-\ln Z(\boldsymbol{\theta}) \approx \min_{\boldsymbol{\tau} \in \mathcal{C}} F(\boldsymbol{\tau}) \qquad \text{and} \qquad -\ln Z(\boldsymbol{x}, \boldsymbol{\theta}) \approx \min_{\boldsymbol{\tau}_x \in \mathcal{C}} F_{\boldsymbol{x}}(\boldsymbol{\tau}_{\boldsymbol{x}}),$$

where $F_{\boldsymbol{x}}(\boldsymbol{\tau}_{\boldsymbol{x}})$ is the Bethe free energy of equation 2 with $\boldsymbol{x}$ clamped to particular values. That is, $F_{\boldsymbol{x}}$ does not consider marginals corresponding to settings of $\mathbf{x}$ that do not agree with $\boldsymbol{x}$, and thus $\boldsymbol{\tau}_{\boldsymbol{x}}$ will in general be smaller than $\boldsymbol{\tau}$. In particular, to continue the example of Figure 1 (c) from the previous section, where all variables are assumed to be binary, $\boldsymbol{\tau}_{\boldsymbol{x}}$ will be in $[0,1]^{32}$ instead of $[0,1]^{40}$, since for each variable in $x_1, \ldots, x_4$ we ignore the two marginals corresponding to the unobserved value.

We may then define the following loss, as an approximation (but neither an upper nor lower bound) of equation 1:

$$\begin{aligned} \ell_F(\boldsymbol{\theta}) &= \min_{\boldsymbol{\tau}_{\boldsymbol{x}} \in \mathcal{C}} F_{\boldsymbol{x}}(\boldsymbol{\tau}_{\boldsymbol{x}}) - \min_{\boldsymbol{\tau} \in \mathcal{C}} F(\boldsymbol{\tau}) \\ &\approx -\ln Z(\boldsymbol{x}, \boldsymbol{\theta}) + \ln Z(\boldsymbol{\theta}), \end{aligned} \tag{3}$$

and thus we arrive at the following saddle-point learning problem:

$$\begin{aligned} \min_{\boldsymbol{\theta}} \ell_F(\boldsymbol{\theta}) &= \min_{\boldsymbol{\theta}} [\min_{\boldsymbol{\tau}_{\boldsymbol{x}} \in \mathcal{C}} F_{\boldsymbol{x}}(\boldsymbol{\tau}_{\boldsymbol{x}}) - \min_{\boldsymbol{\tau} \in \mathcal{C}} F(\boldsymbol{\tau})] \\ &= \min_{\boldsymbol{\theta}, \boldsymbol{\tau}_{\boldsymbol{x}} \in \mathcal{C}} \max_{\boldsymbol{\tau} \in \mathcal{C}} [F_{\boldsymbol{x}}(\boldsymbol{\tau}_{\boldsymbol{x}}) - F(\boldsymbol{\tau})]. \end{aligned} \tag{4}$$

3.1 CONSTRAINED OPTIMIZATION

While LBP can be used to find pseudo-marginals representing fixed points of F and $F_{\boldsymbol{x}}$, it is somewhat unappealing in the context of deep generative modeling, primarily because it is an iterative message-passing algorithm often requiring multiple rounds for convergence, and where the order in which messages are passed appears to significantly impact results (Yuille, 2002).

Instead, we propose to *predict* approximate minimizers of the Bethe free energy using trainable inference networks $f(\mathcal{G};\, \boldsymbol{\phi})$ and $f_x(\mathcal{G}, \boldsymbol{x};\, \boldsymbol{\phi}_x)$, which will compute approximate minimizers of F and $F_{\boldsymbol{x}}$, respectively.[2] These inference networks, which are similar to graph neural networks (Scarselli et al., 2009; Li et al., 2015; Kipf & Welling, 2016; Yoon et al., 2018) but somewhat simpler, will attempt to find minimizers living in the constraint set $\mathcal{C}$, which, as noted in Section 2.2, consists of vectors $\boldsymbol{\tau}$ containing pseudo-marginals which are positive and sum to one, and which respect local consistency. We address the parameterization of the inference networks, and how they handle these constraints on the pseudo-marginals below.

Equality Constraints It is clear that the constraints that pseudo-marginals sum to one and that they exhibit local consistency are linear constraints on $\boldsymbol{\tau}$, and so they can be expressed as $\boldsymbol{A}\boldsymbol{\tau} = \boldsymbol{b}$, where $\boldsymbol{A} \in \{0, 1\}^{m \times n}$ consists of m linearly independent constraint rows and and n is the length of $\boldsymbol{\tau}$. It is standard in linear-equality constrained optimization to optimize over the subspace defined by these linear constraints instead of the original optimization variable $\boldsymbol{\tau}$ (Boyd & Vandenberghe, 2004). In particular, given a feasible point $\hat{\boldsymbol{\tau}}$ and a basis $\boldsymbol{V} \in \mathbb{R}^{n \times n-m}$ for the null space of $\boldsymbol{A}$, any solution can be written as $\boldsymbol{V}\boldsymbol{u} + \hat{\boldsymbol{\tau}}$, and so we may optimize over $\boldsymbol{u}$ instead of $\boldsymbol{\tau}$.

When using inference networks to compute minimizers, however, it is more natural to have these networks output vectors of length n (i.e., the size of $\boldsymbol{\tau}$, which is linear in the factors of the MRF), rather than of size $n - m$, which depends on the number of constraints. Accordingly, we may equivalently write minimizers as $\boldsymbol{V}\boldsymbol{V}^{+}\boldsymbol{\rho} + \hat{\boldsymbol{\tau}}$, where $\boldsymbol{\rho} \in \mathbb{R}^n$, and $\boldsymbol{V}^{+}$ is the Moore-Penrose pseudoinverse of $\boldsymbol{V}$. In particular, $\boldsymbol{V}\boldsymbol{V}^{+}$ is an orthogonal projection on the range of $\boldsymbol{V}$, a basis for the null space of $\boldsymbol{A}$. Thus, given an inference network f that computes vectors in $\mathbb{R}^n$, the vector $(\boldsymbol{V}\boldsymbol{V}^{+} f(\mathcal{G};\, \boldsymbol{\phi}) + \hat{\boldsymbol{\tau}}) \in \mathbb{R}^n$ will satisfy the equality constraints imposed by $\mathcal{C}$, assuming $\hat{\boldsymbol{\tau}}$ does.

Positivity Constraints In order to keep elements of the predicted pseudo-marginals positive, we simply impose a penalty during training on predicting a pseudo-marginal with non-positive elements. (If, at evaluation time, a predicted pseudo-marginal is non-positive, we set it to a small positive constant). We found a linear penalty on non-positive values to work well: given a vector $\boldsymbol{\rho} \in \mathbb{R}^n$, we define the penalty function $C(\boldsymbol{\rho}) = -\frac{1}{n}\sum_{i=1}^{n} \min\{\rho_i, 0\}$. We thus arrive at the following training objective:

$$\min_{\boldsymbol{\theta}, \boldsymbol{\phi}} \max_{\boldsymbol{\phi}_x} \big[F_{\boldsymbol{x}}(\boldsymbol{P}_x f_x(\boldsymbol{x}, \mathcal{G};\, \boldsymbol{\phi}_x) + \hat{\boldsymbol{\tau}}_x) + \lambda C(\boldsymbol{P}_x f_x(\boldsymbol{x}, \mathcal{G};\, \boldsymbol{\phi}_x) + \hat{\boldsymbol{\tau}}_x) \\ - F(\boldsymbol{P} f(\mathcal{G};\, \boldsymbol{\phi}) + \hat{\boldsymbol{\tau}}) - \lambda C(\boldsymbol{P} f(\mathcal{G};\, \boldsymbol{\phi}) + \hat{\boldsymbol{\tau}}) \big], \tag{5}$$

where $\boldsymbol{P}$ and $\boldsymbol{P}_x$ are the orthogonal projections $\boldsymbol{V}\boldsymbol{V}^{+}$ defined by the constraints on $\boldsymbol{\tau}$ and $\boldsymbol{\tau}_{\boldsymbol{x}}$, respectively.

4 EXPERIMENTS

In order to investigate how the approximate inference approach outlined above compares with other popular approaches to approximate inference, we will use approximate inference to learn in a setting in which we

[2] While we also experimented with obtaining approximate minimizers with gradient descent, an approach first proposed by Welling & Teh (2001) for the purpose of performing inference (but not learning), we found this approach to be both slower and less performant than using inference networks.

can tractably compute log marginal likelihoods. We consider in particular learning 2nd and 3rd order neural HMMs, as in Figure 1, on text, using various flavors of amortized variational inference as well as the Bethe-based objective ℓ_F introduced above. (Note that Bethe based objectives are inexact for loopy MRFs, such as the HMM-style MRFs with long-distance pairwise factors in Figure 1 (c)).

Because the Bethe objective is only interesting in the case of undirected models (e.g., Figure 1 (c)), we will strictly speaking be comparing slightly different model classes. However, this comparison likely favors the directed models, and as our experiments confirm both model classes can obtain similar held-out perplexities when learned with exact inference. We accordingly begin by briefly outlining how the neural HMMs and their associated inference networks are parameterized. To simplify the notation somewhat, in what follows we will view HMMs as parameterizing a joint distribution over a sequence of T discrete observations $x_{1:T}$ and a sequence of T discrete latent variables $z_{1:T}$, where each observation z_t takes one of V values, and each latent z_t takes one of K values.

4.1 MODELS

Neural Directed HMM We parameterize the HMM's emission distribution $P(x_t \mid z_t = k)$, as $\mathrm{softmax}(\boldsymbol{W}\,\mathrm{LayerNorm}(\boldsymbol{e}_k + \mathrm{MLP}(\boldsymbol{e}_k)))$, where $\boldsymbol{e}_k \in \mathbb{R}^d$ is an embedding corresponding to the k'th discrete value z_t can take on, $\boldsymbol{W} \in \mathbb{R}^{V \times d}$ is a word embedding matrix with a row for each word in the vocabulary, and layer normalization (Ba et al., 2016) is used to stabilize training. We parameterize the transition distribution $P(z_t \mid z_{t-1} = k_1, \ldots, z_{t-M} = k_M)$ similarly, as $\mathrm{softmax}(\boldsymbol{U}\,\mathrm{LayerNorm}([\boldsymbol{e}_{k_1}; \ldots; \boldsymbol{e}_{k_M}] + \mathrm{MLP}([\boldsymbol{e}_{k_1}; \ldots; \boldsymbol{e}_{k_M}])))$, where $\boldsymbol{U} \in \mathbb{R}^{K \times MK}$ and the $\boldsymbol{e}_k$ are shared with the emission parameterization.

Mean Field Inference Network When performing amortized variational inference with a mean field posterior, we obtain approximate posteriors $q(z_t \mid x_{1:T})$ for each timestep t as $\mathrm{softmax}(\boldsymbol{Q}\boldsymbol{h}_t)$, where $\boldsymbol{h}_t \in \mathbb{R}^{d_2}$ is the output of a bidirectional LSTM (Hochreiter & Schmidhuber, 1997; Graves et al., 2013) run over the observations $x_{1:T}$, and $\boldsymbol{Q} \in \mathbb{R}^{K \times d_2}$.

Structured Inference Network Instead of assuming the approximate posterior $q(z_{1:T} \mid x_{1:T})$ factorizes independently over timesteps posteriors as in mean field, we can assume it is given by the posterior of a first-order (and thus more tractable) HMM. We parameterize this inference HMM identically to the neural directed HMM above, except that it conditions on the observed sequence $x_{1:T}$ by concatenating the averaged hidden states of a bidirectional LSTM run over the sequence onto the $\boldsymbol{e}_k$.

Undirected Neural HMM We parameterize the emission factors $\Psi_\alpha(x_t, z_t)$ as locally normalized distributions, in exactly the same way as the neural directed HMM above. In order to fairly compare with the directed HMM, the transition factors $\Psi_\alpha(z_s = k_1, z_t = k_2)$ are homogeneous (i.e., independent of the timestep), and are given by $\boldsymbol{r}_{k_2}^{\mathsf{T}}\,\mathrm{LayerNorm}([\boldsymbol{a}_{|t-s|}; \boldsymbol{e}_{k_1}] + \mathrm{MLP}([\boldsymbol{a}_{|t-s|}; \boldsymbol{e}_{k_1}]))$, where $\boldsymbol{a}_{|t-s|}$ is the embedding vector corresponding to factors relating two nodes that are $|t-s|$ steps apart, and where $\boldsymbol{e}_{k_1}$ and $\boldsymbol{r}_{k_2}$ are again discrete state embedding vectors.

Bethe Inference Networks In the case of sequential models like HMMs, it is fairly simple to parameterize $f(\mathcal{G};\, \boldsymbol{\phi})$ and $f_x(\mathcal{G}, \boldsymbol{x};\, \boldsymbol{\phi}_x)$. For $f(\mathcal{G};\, \boldsymbol{\phi})$ we form an embedding for each factor $\Psi_\alpha(z_s, z_t)$ by concatenating embedding vectors corresponding to the s'th and t'th timesteps with the K^2 log potential values $\log \Psi_\alpha(z_s = k_1, z_t = k_2)$ (as given above), and then run a bidirectional LSTM over these factor-embeddings, ordered in ascending order of z_s. We then predict pseudo-marginals for each factor with a linear layer applied to the LSTM output. The parameterization of $f_x(\mathcal{G}, \boldsymbol{x};\, \boldsymbol{\phi}_x)$ is similar, except we also concatenate the log observation potentials $\log \Psi_\alpha(x_s, z_s)$ and $\log \Psi_\alpha(x_t, z_t)$ onto the embedding of factor $\Psi_\alpha(z_s, z_t)$ before feeding it to the LSTM.

Table 1: Rows from top to bottom give the performance of directed HMMs learned with a mean-field VAE objective, with a mean-field IWAE objective with 5 and 10 samples (resp.), with a 1st order HMM VAE objective, and with the exact log marginal. Approximate objectives use REINFORCE-like gradient estimators. ELBOs (or IWAE bounds) are reported as exponentiated, token-averaged negative ELBO to be comparable with PPL.

Objective	Train PPL	Train ELBO	Val. PPL	Val. ELBO
2nd Order HMM				
Mean Field VAE + BL	356.278	7493.916	348.055	7318.822
Mean Field IWAE, $L = 5$	346.241	7406.263	338.069	7224.112
Mean Field IWAE, $L = 10$	335.622	7270.175	328.424	7087.252
1st Order HMM VAE	279.894	291.144	286.399	298.078
Exact	145.275	N/A	163.029	N/A
3rd Order HMM				
Mean Field VAE + BL	356.475	7512.892	350.983	7382.303
Mean Field IWAE, $L = 5$	357.002	7545.066	346.573	7290.135
Mean Field IWAE, $L = 10$	333.796	7251.066	335.241	7273.600
1st Order HMM VAE	268.911	276.495	270.249	274.118
Exact	139.261	N/A	157.410	N/A

4.2 Methods

We trained the models described above under both VAE-style and Bethe objectives (elaborated upon below) on 16,737 sentences of length at most 20 from the Penn Treebank corpus (Marcus et al., 1993), and evaluated them on a held out sample of 1,585 sentences. Models and objectives were evaluated in terms of their *true* perplexity on the held out data, which can be computed reasonably efficiently with dynamic programs. We found all models and objectives to be fairly sensitive to hyperparameters and random seeds, and so we report the best results obtained (in terms of held out true perplexity) by each model and objective after a random search over hyperparameter and seed settings.

The saddle-point objective ℓ_F was optimized by alternating a maximization step wrt $\boldsymbol{\phi}$, a minimization step wrt $\boldsymbol{\phi}_x$, and a minimization step wrt $\boldsymbol{\theta}$. Projection matrices $\boldsymbol{P}$ for each graph structure were pre-calculated and stored, and feasible pseudo-marginals $\hat{\boldsymbol{\tau}}$ can be obtained by assigning all marginals to be uniform. Code for duplicating all experiments is available at `https://hidden.for.anonymity`.

4.3 Results and Discussion

We begin with the results obtained by maximizing the *true* log marginal likelihood of the training data under both the directed and undirected models, by backpropagating gradients through the relevant dynamic programs. These results establish how well our models perform under exact inference, and are shown in the last row of each subtable in Tables 1 and 2. We see that perplexities are roughly comparable between the directed and undirected models when trained with exact inference, with the directed model obtaining an improvement of about 15 points in perplexity in the 3rd order case.

We now consider the remaining results of Table 1, where the directed HMM models are trained with approximate inference. In the first row of each subtable there, we show the result of maximizing the ELBO using a mean field posterior approximation and the REINFORCE (Williams, 1992) gradient estimator, with

Table 2: Rows from top to bottom give the performance of undirected HMMs learned with the ℓ_F objective but exact marginals, the ℓ_F objective with approximate marginals given by inference networks, and the exact log marginal. Values of ℓ_F are reported as exponentiated, token-averaged negative ELBO to be comparable with PPL.

Objective	Train PPL	Train ℓ_F	Val. PPL	Val. ℓ_F
2nd Order HMM				
ℓ_F + exact marginals	139.896	113.196	151.781	125.100
ℓ_F + f, f_x	256.120	303.531	266.420	332.522
Exact	151.942	N/A	161.55	N/A
3rd Order HMM				
ℓ_F + exact marginals	159.187	155.038	170.071	152.881
ℓ_F + f, f_x	247.607	262.740	259.458	175.974
Exact	161.728	N/A	173.374	N/A

an input-dependent baseline to reduce variance (Mnih & Gregor, 2014). The results are quite poor, with this approximate inference scheme leading to a gain of almost 200 points in perplexity over exact inference. Using the tighter IWAE (Burda et al., 2015) objectives improves performance slightly in all cases, though the most dramatic performance improvement comes from using a first-order HMM posterior in maximizing the ELBO, which can be sampled from exactly using quantities calculated with the forward algorithm (Rabiner, 1989; Chib, 1996; Scott, 2002; Zucchini et al., 2016). While these results are encouraging, note that in general we may not have an exact dynamic program for sampling from a lower-order structured model, and that moreover we still appear to incur a perplexity penalty of more than 100 points over exact inference.

The second row of each subtable in Table 2 contains the results of optimizing ℓ_F as a saddle point problem. While it too underperforms exact inference by approximately 100 points in perplexity, somewhat remarkably it manages to consistently outperform the best approximate inference results for the directed models by approximately ten points in perplexity. The first row of each subtable in Table 2 attempts to determine whether the jump in perplexity when moving to the ℓ_F objective is due to the approximate inference or to the approximate objective, by minimizing the ℓ_F objective using the *exact* marginals, as calculated by a dynamic program. (Note that this is not equivalent to the negative log marginal likelihood, since the factor graphs are loopy). Interestingly, we see that this performs at least as well as the exact objective, suggesting that, at least for HMM models, the ℓ_F objective is reasonable, and approximate inference remains the problem.

We note that there are several drawbacks to the proposed approach, however. In particular, Table 2 shows that in practice ℓ_F indeed can over- or under-estimate perplexity. Moreover, while ELBO values are not perfectly correlated with their corresponding true perplexities, values of ℓ_F seem even less correlated, which necessitates finding correlated proxies of perplexity that may be monitored during training.

5 Conclusion

We have presented an objective for learning latent-variable MRFs based on the Bethe approximation to the partition function, which can often be efficiently evaluated and requires no sampling. This objective leads to slightly better held-out perplexities than other approximate inference methods when learning neural HMMs. Future work will examine scaling the proposed method to larger, non-sequential MRFs, and whether ℓ_F-like objectives can be made to better correlate with the true perplexity.

REFERENCES

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Hans A Bethe. Statistical theory of superlattices. *Proceedings of the Royal Society of London. Series A-Mathematical and Physical Sciences*, 150(871):552–575, 1935.

Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance Weighted Autoencoders. In *Proceedings of ICLR*, 2015.

Siddhartha Chib. Calculating posterior distributions and modal estimates in markov mixture models. *Journal of Econometrics*, 75(1):79–97, 1996.

Brendan J Frey, Frank R Kschischang, Hans-Andrea Loeliger, and Niclas Wiberg. Factor graphs and algorithms. In *Proceedings of the Annual Allerton Conference on Communication Control and Computing*, volume 35, pp. 666–680. University of Illinois, 1997.

Matthew Gormley and Jason Eisner. Structured belief propagation for nlp. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: Tutorials*, pp. 9–10, 2014.

Matthew R Gormley. *Graphical Models with Structured Factors, Neural Factors, and Approximation-Aware Training*. PhD thesis, Johns Hopkins University, 2015.

Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with deep bidirectional lstm. In *2013 IEEE workshop on automatic speech recognition and understanding*, pp. 273–278. IEEE, 2013.

Sepp Hochreiter and J´urgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9:1735–1780, 1997.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical Reparameterization with Gumbel-Softmax. In *Proceedings of ICLR*, 2017.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pp. 694–711. Springer, 2016.

Michael Jordan, Zoubin Ghahramani, Tommi Jaakkola, and Lawrence Saul. Introduction to Variational Methods for Graphical Models. *Machine Learning*, 37:183–233, 1999.

Lukasz Kaiser, Samy Bengio, Aurko Roy, Ashish Vaswani, Niki Parmar, Jakob Uszkoreit, and Noam Shazeer. Fast Decoding in Sequence Models Using Discrete Latent Variables. In *Proceedings of ICML*, 2018.

Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *Proceedings of ICLR*, 2014.

Thomas N Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.

Frank R Kschischang, Brendan J Frey, Hans-Andrea Loeliger, et al. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2):498–519, 2001.

Volodymyr Kuleshov and Stefano Ermon. Neural variational inference and learning in undirected graphical models. In *Advances in Neural Information Processing Systems*, pp. 6734–6743, 2017.

Jason Lee, Elman Mansimov, and Kyunghyun Cho. Deterministic Non-Autoregressive Neural Sequence Modeling by Iterative Refinement. In *Proceedings of EMNLP*, 2018.

Chongxuan Li, Chao Du, Kun Xu, Max Welling, Jun Zhu, and Bo Zhang. Adversarial variational inference and learning in markov random fields. *arXiv preprint arXiv:1901.08400*, 2019.

Yujia Li, Daniel Tarlow, Marc Brockschmidt, and Richard Zemel. Gated graph sequence neural networks. *arXiv preprint arXiv:1511.05493*, 2015.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *Proceedings of ICLR*, 2017.

Mitchell Marcus, Beatrice Santorini, and Mary Marcinkiewicz. Building a Large Annotated Corpus of English: the Penn Treebank. *Computational Linguistics*, 19:331–330, 1993.

Ofer Meshi, Ariel Jaimovich, Amir Globerson, and Nir Friedman. Convexifying the bethe free energy. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pp. 402–410. AUAI Press, 2009.

Andriy Mnih and Danilo J. Rezende. Variational Inference for Monte Carlo Objectives. In *Proceedings of ICML*, 2016.

Andryi Mnih and Karol Gregor. Neural Variational Inference and Learning in Belief Networks. In *Proceedings of ICML*, 2014.

Judea Pearl. Fusion, propagation, and structuring in belief networks. *Artificial intelligence*, 29(3):241–288, 1986.

Lawrence R. Rabiner. A Tutorial on Hidden markov Models and Selected Applications in Speech Recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Danilo J. Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of ICML*, 2014.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009.

Steven L Scott. Bayesian methods for hidden markov models: Recursive computing in the 21st century. *Journal of the American Statistical Association*, 97(457):337–351, 2002.

Vivek Srikumar, Gourab Kundu, and Dan Roth. On amortizing inference cost for structured prediction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 1114–1124. Association for Computational Linguistics, 2012.

Ke Tran, Yonatan Bisk, Ashish Vaswani, Daniel Marcu, and Kevin Knight. Unsupervised Neural Hidden Markov Models. In *Proceedings of the Workshop on Structured Prediction for NLP*, 2016.

Lifu Tu and Kevin Gimpel. Learning approximate inference networks for structured prediction. In *ICLR*, 2018.

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

Adrian Weller and Tony Jebara. Approximating the bethe partition function. In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pp. 858–867. AUAI Press, 2014.

Adrian Weller, Kui Tang, David Sontag, and Tony Jebara. Understanding the bethe approximation: when and how can it go wrong? In *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*, pp. 868–877. AUAI Press, 2014.

Max Welling and Yee Whye Teh. Belief optimization for binary networks: A stable alternative to loopy belief propagation. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pp. 554–561. Morgan Kaufmann Publishers Inc., 2001.

Ronald J. Williams. Simple Statistical Gradient-following Algorithms for Connectionist Reinforcement Learning. *Machine Learning*, 8, 1992.

Alan S Willsky, Erik B Sudderth, and Martin J Wainwright. Loop series and bethe variational bounds in attractive graphical models. In *Advances in neural information processing systems*, pp. 1425–1432, 2008.

Jonathan S Yedidia, William T Freeman, and Yair Weiss. Generalized belief propagation. In *Advances in neural information processing systems*, pp. 689–695, 2001.

Jonathan S Yedidia, William T Freeman, and Yair Weiss. Understanding belief propagation and its generalizations. *Exploring artificial intelligence in the new millennium*, 8:236–239, 2003.

KiJung Yoon, Renjie Liao, Yuwen Xiong, Lisa Zhang, Ethan Fetaya, Raquel Urtasun, Richard Zemel, and Xaq Pitkow. Inference in probabilistic graphical models by graph neural networks. *arXiv preprint arXiv:1803.07710*, 2018.

Alan L Yuille. Cccp algorithms to minimize the bethe and kikuchi free energies: Convergent alternatives to belief propagation. *Neural computation*, 14(7):1691–1722, 2002.

Walter Zucchini, Iain L MacDonald, and Roland Langrock. *Hidden Markov models for time series: an introduction using R*. Chapman and Hall/CRC, 2016.