Final project:

Latent Dependency Structure + Flow IAF optimization on Variational Autoencoder

Abstract

We propose a method that combines both learning a dependency structure between latent variables and adding an Inverse autoregressive flow. This provides a flexible variational inference of posteriors over latent variables and a probabilistic graphical structure in latent space on a deep latent variable model. In practice, we model a variational autoencoder (VAE) and learn a flexible dependency structure of the latent variables with an IAF normalizing flow. In the inference model we produce expectations over latent variable structures and incorporate top-down and bottom-up reasoning over latent variable values, then a flow network that consists of a chain of invertible transformations which are based on autoregressive neural network is added. The model is validated on MNIST experiments and compared to separate methods of VAE + IAF and VAE with latent dependency structure.

Introduction

Provided large amount of data and a good network architecture, deep latent variable models such as Variational Autoencoders (VAE) are very effective at learning compressed representations that can produce highly realistic content (images, text, sound, etc). Using Neural networks for both inference network and generative model to model the latent data distributions. Although successful, these models usually initialize a gaussian prior distribution with the assumption that each latent variable is sampled independently. Ignoring the dependencies between variables can limit the model's flexibility to fit the data.

One can incorporate structured dependencies into all phases of the forward process, particularly one can add dependencies by constructing a hierarchical latent representation with empirical priors that are connected to parent latent variables.

Another general approach to improving VAE performance is to build flexible inference networks with normalizing flows by learning a mapping from a simple posterior distribution to a more complex posterior. This can be done using inverse autoregressive flow.

In this work, we propose to improve VAE performance by both adding hierarchical latent dependencies and building an inference network with normalizing flow. We use the methods from the paper "VARIATIONAL AUTOENCODERS WITH JOINTLY OPTIMIZED LATENT DEPENDENCY STRUCTURE"[1] that suggests learning these latent dependencies, rather than using predefined models with potentially limited performance, and the paper "Improved Variational Inference with Inverse Autoregressive Flow"[2] that suggests a new type of normalizing flow framework, inverse autoregressive flow (IAF), which improves on the diagonal Gaussian approximate posteriors and scales well to high-dimensional latent space.

By combining elements from both papers, we benefit from both improvements of prior and posterior distributions on VAE, which can produce better results and a more flexible latent space.

Background

Variational Autoencoders with jointly optimized latent dependency structure

We will refer to this model as "VAE with dependency structure".

This paper implements a Variational Autoencoder model with learned graphical structures on the latent space. A set of binary global variables ($c \sim \{0,1\}$) is introduced to gate the latent dependencies. The variable c specifies the presence (c=1) or absence (c=0) of a latent dependency between nodes. The prior on the latent variables is expressed by (Eq.6 in paper):

$$p_\theta(z|c) = \prod_{n=1}^{N} p_\theta(z_n|z_{pa(n)}, c_{pa(n),n})$$

where $c_{pa(n),n}$ denotes the gate variables associated with the dependencies between node z_n and its parents, $z_{pa(n)}$. Note that $z_{pa(n)}$ denotes the set of all possible parents of node z_n in the fullyconnected DAG, i.e. $z_{pa(n)} = \{z_{n+1}, . . . , z_N \}$.

The distribution $p_\theta(z_n|z_{pa(n)}, c_{pa(n),n})$ is given by the Gaussian density with the parameters $\hat{\psi}_n = (\hat{\mu}_n, \hat{\Sigma}_n)$. These parameters are obtained by recursively multiplying samples of $z_{pa(n)}$ (parents of node z_n) with the corresponding gating parameters $c_{pa(n),n}$ and inputting a concatenation of the multiplication into a MLP(Top Down)_n layer, predicting $\hat{\psi}_n$. The top-down recursion starts at the root node $z_N \sim p(z_N) = N (0, I)$.

Illustration of the model taken from the paper [1]:

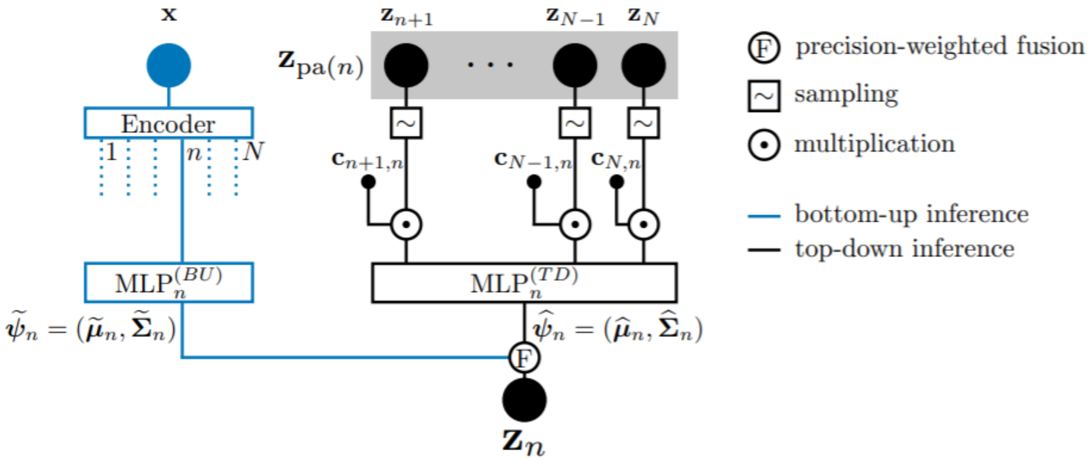


Figure 2: **Local Distributions.** We illustrate the parametrization of a local variable $\mathbf{z}_n$ in our structured representation. The local prior (Eq. (6)) is defined in terms of a top-down process (in black) predicting the node's parameters $\widehat{\psi}_n$ from a gate-modulated sample of $\mathbf{z}_n$'s parents $\mathbf{z}_{\text{pa}(n)}$. The local approximate posterior (Eq. (7)) additionally performs a precision-weighted fusion of these parameters with the result of a bottom-up process using a node-specific MLP to predict input-conditioned parameters $\widetilde{\psi}_n$ from a generic encoding of $\mathbf{x}$.

The approximate posterior, $q_\phi(z|x, c)$, must approximate $p_\theta(z|x, c)$.

The approximate posterior is expressed as (Eq.7 in paper):

$$q_{\emptyset}(z|x,c) = \prod_{n=1}^{N} q_{\emptyset}(z_n|x, z_{pa(n),}c_{pa(n)}$$

The prediction for the parameters of the distribution $q_{\emptyset}(z_n|x, z_{pa(n),}c_{pa(n)})$ are predicted by a weighted fusion of the top-down prediction $\hat{\psi}$n and a bottom-up prediction with gaussian density parameters $\tilde{\psi}_n \;=\; (\tilde{\mu}_n, \tilde{\sum}_n)$. The bottom-up is obtained by encoding x into a generic feature that is used as an input to a MLP(Bottom Up)n, predicting $\tilde{\psi}_n$.

Improved Variational Inference with Inverse Autoregressive Flow

We will refer to this model as "VAE + IAF".

Normalizing Flow frameworks help build flexible posterior distributions through an iterative procedure. The general idea is to start off with an initial random variable z_0, with a relatively simple distribution with known (and computationally cheap) probability density function, and then apply a chain of invertible parameterized transformations f_t, such that the last iterate z_t has a more flexible distribution.

$$z_0 \sim q(z_0|x), \quad z_t = f_t(z_{t-1,}x) \quad \forall t = 1 \dots . T$$

The log posterior can be computed as long as the Jacobian determinant of each of the transformations can be computed. And is given by the following equation:

$$\log q(z_T|x) = \log q(z_0|x) - \sum_{t=1}^{T} \log\det\left|\frac{d_{z_t}}{d_{z_{t-1}}}\right|$$

In this paper the normalizing flow is an autoregressive flow. The initial encoder outputs μ_0 and $\sum_0$ and an additional output h, which serves as an additional input to each step in the flow.

A random sample is drawn from $\varepsilon \sim N(0, I)$ and the chain is initialized:

$$z_0 = \mu_0 + \textstyle\sum_0 \odot \varepsilon$$

Illustration of the model taken from the paper [2]:

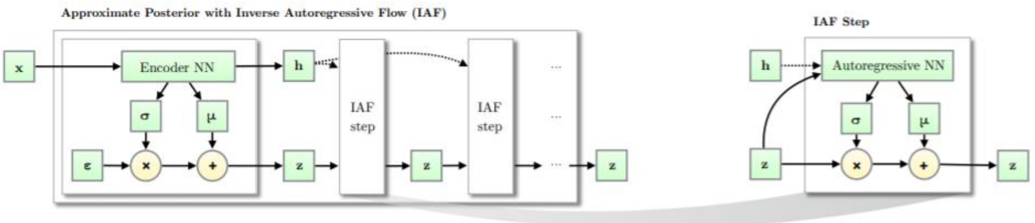


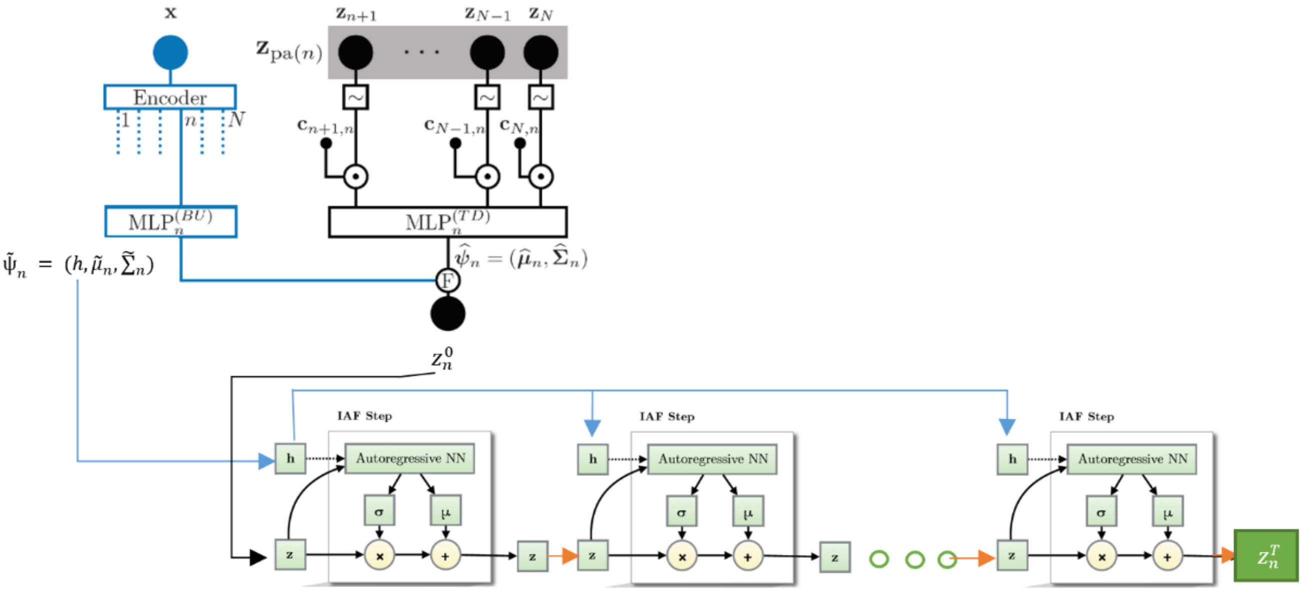
Figure 2: Like other normalizing flows, drawing samples from an approximate posterior with Inverse Autoregressive Flow (IAF) consists of an initial sample **z** drawn from a simple distribution, such as a Gaussian with diagonal covariance, followed by a chain of nonlinear invertible transformations of **z**, each with a simple Jacobian determinants.

The flow consists of a chain of T of the following transformations:

$$z_t = \mu_t + \textstyle\sum_t \odot z_{t-1}$$

Method

We propose to combine elements of both papers by using the same encoder suggested in the VAE with dependency structure model and adding an additional output h_n from the encoders $MLP(BU)_n$. Then we use the inference fussion output, z_{n0}, and h_n as the input to the IAF layers from the VAE + IAF model. The output of the IAF layers z_{nT} is added to the next nodes parents $z_{pa(n+1),}$ as can be seen from the following structure:



The root node z_N is initially obtained only by the bottom-up parameter prediction, $\tilde{\psi}_n$, as there are no parent nodes to consider, it is then passed through the IAF layers.

To optimize our model, we optimize a monte Carlo estimation, of a combined loss function:

$$ELBO = E_{p(c)}\left[E_{q(z_T|x,c)}[\log p(x|z_T) + \log(p(z_T|c) - \log q(z_T|x)]\right]$$

$$ELBO = E_{p(c)}\left[E_{q(z_T|x,c)}\left[\log p(x|z_T) + \log(p(z_T|c) - \log q(z_0|x) + \sum_{t=1}^{T} log\left|\frac{dz_T}{dz_{T-1}}\right|\right]\right]$$

We also find the log likelihood for 100 samples (M=100) on the test set using the monte-carlo estimation.

$$p(x) = \int_z p(x,z)\,dz = \int_z p(x,z)\frac{q(z|x)}{q(z|x)}dz = E_{z\sim q(z|x)}\left(\frac{p(x|z)P(z)}{q(z|x)}\right)$$

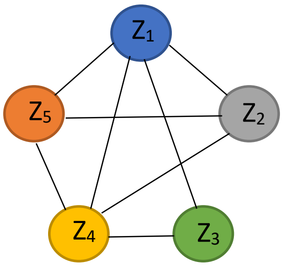
$$\xRightarrow{\substack{Monte-Carlo\\ estimation}} \frac{1}{M}\sum_{j=1}^{M}\frac{p(x_i|z_j)P(z_j)}{q(z_j|x_i)}$$

The model is tested on the binarized MNIST dataset. We train the model with 5 nodes. Initially we train the model for 200 epochs until the gating parameters converge. We then use the learned c gates as fixed parameters in the model and retrain the model for 600 epochs until convergence for the specific trained latent dependency structure.

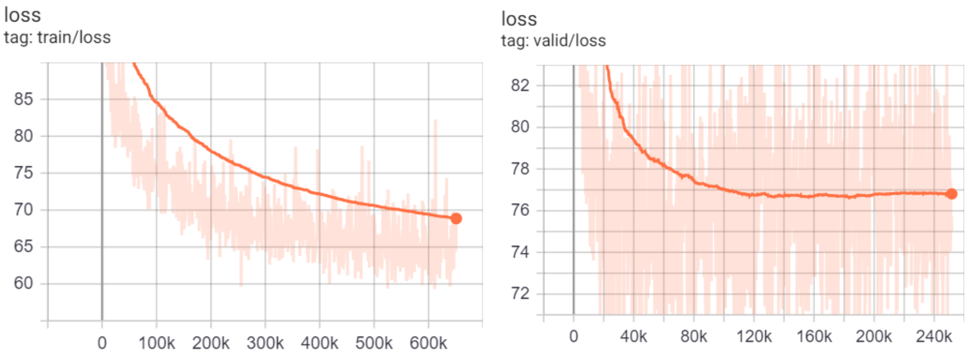
Results

We evaluate our results against two other methods. The first is an implementation of the VAE with dependency structure model, where we optimize a Monte-Carlo estimated loss rather than the suggested analytical approach to get a more uniform comparison to our model which is also optimized with a Monte-Carlo estimated loss. The second comparison is to an implementation of the VAE + IAF model. Note that we evaluate against results we obtained for 600 epochs for a code implementation, these results are different to the recorded results shown in the relative papers.

After convergence (200 epochs), the learned gate parameters c has dependencies between all nodes apart from the connection Z_2 - Z_3, as seen in the following graph:



Graph of train and validation ELBO loss function to steps (~ 600 epochs are shown):



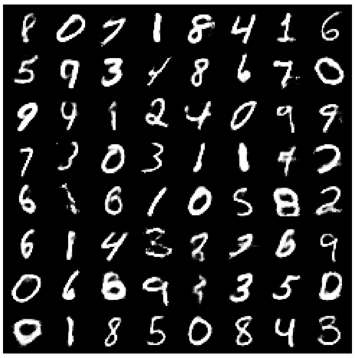
Results on the MNIST test dataset:

Method	ELBO	Log Likelihood
VAE + IAF	-102.6	-95.3
Monte Carlo VAE with dependency structure	-79.4	-73.7
Our combined	-77.7	-71.87

From the graphs we can see the ELBO loss function decreases with epochs for both train and validation, although with some fluctuations that might be due to doing a monte-carlo estimation with 1 sample. From the test results we can see that for both ELBO and Log

Likelihood values our model improves on the results obtained from the other methods presented, with ELBO of -77.7 and a Log Likelihood of -71.87. This indicates that the added flexibility to the latent space helped get better results.

Sampled results after 600 epochs for our "combined" method:



Bibliography:

1. Jiawei He & Yu Gong, et al, 2019, VARIATIONAL AUTOENCODERS WITH JOINTLY OPTIMIZED LATENT DEPENDENCY STRUCTURE.
2. Diederik P. Kingma, Tim Saliman, et al, 2017, Improved Variational Inference with Inverse Autoregressive Flow.