Semi-supervised Learning by Latent Space Energy-Based Model of Symbol-Vector Coupling

Bo Pang\textsuperscript{1} \quad Erik Nijkamp\textsuperscript{1} \quad Jiali Cui\textsuperscript{2} \quad Tian Han\textsuperscript{2} \quad Ying Nian Wu\textsuperscript{1}

\textsuperscript{1}University of California, Los Angeles \quad \textsuperscript{2}Stevens Institute of Technology

\{bopang, enijkamp\}@ucla.edu \quad \{jcui7, than6\}@stevens.edu \quad ywu@stat.ucla.edu

Abstract

This paper proposes a latent space energy-based prior model for semi-supervised learning. The model stands on a generator network that maps a latent vector to the observed example. The energy term of the prior model couples the latent vector and a symbolic one-hot vector, so that classification can be based on the latent vector inferred from the observed example. In our learning method, the symbol-vector coupling, the generator network and the inference network are learned jointly. Our method is applicable to semi-supervised learning in various data domains such as image, text, and tabular data. Our experiments demonstrate that our method performs well on semi-supervised learning tasks.

1 Introduction

Generative modeling and likelihood-based learning is a principled framework for unsupervised and semi-supervised learning. The energy-based model (EBM) \cite{29}, being a generative version of a discriminator, is particularly convenient for semi-supervised learning. A notable recent paper is Grathwohl et al. \cite{13}, which proposed the joint energy-based model (JEM) and applied it to classification. In an extension \cite{46}, it was applied to semi-supervised learning. Earlier papers that touch on the relationship between modern EBM and classifier parametrized by deep networks \cite{28,25} include \cite{6,43,19,27}.

Recently, Pang et al. \cite{36} proposed to learn EBM in latent space, where the EBM serves as a prior model for the latent vector of a generator model that maps the latent vector to the observed example. Both the EBM prior and the generator network can be learned jointly by maximum likelihood or its approximate or variational variants. The latent space EBM has been applied to image modeling, text modeling, and more recently molecule generation \cite{37}. See also the more recent developments \cite{2,42}.

Moving EBM from data space to latent space allows the EBM to stand on an already expressive generator model, and the EBM prior can be considered a correction of the non-informative uniform or isotropic Gaussian prior of the generator model. Due to the low dimensionality of the latent space, the EBM can be parametrized by a very small network, and yet it can capture regularities and rules in the data effectively (and implicitly).

In this paper, we move the formulation of JEM \cite{13} to latent space, so that the latent space EBM prior becomes an associative memory that couples the dense latent vector for generation and the one-hot symbolic vector for classification. Given the inferred latent vector, the conditional distribution of the one-hot vector is a regular softmax classifier based on the latent vector. We learn the symbol-vector coupling, the generator network, and the inference network jointly. Our method is applicable to semi-supervised learning in various data domains such as image, text, and tabular data. In experiments, our method demonstrates either competitive or improved performance compared to previous methods.
The symbol-vector coupling in the latent space is of independent interest in terms of the interaction between symbolic reasoning and distributed representation. We also provide a general theoretical framework in appendix for learning latent EBM in general, where our learning method is a special case.

2 Model and learning

2.1 Model: symbol-vector coupling

Let $x \in \mathbb{R}^D$ be the observed example. Let $z \in \mathbb{R}^d$ be the latent vector. Let $y$ be the symbolic one-hot vector for classification of $K$ categories. $y$ is $K$-dimensional. Our model is defined by

$$p_0(y, z, x) = p_\alpha(y, z)p_\beta(x|z)$$  \hspace{2cm} (1)

where $p_\alpha(y, z)$ is the prior model with parameters $\alpha$, $p_\beta(x|z)$ is the top-down generation model with parameters $\beta$, and $\theta = (\alpha, \beta)$. Given $z$, $y$ and $x$ are independent, i.e., $z$ is sufficient for $y$. After inferring $z$ from $x$, the inference of $y$ is based on $z$, i.e., $z$ is the information bottleneck [41].

The above prior model $p_\alpha(y, z)$ is formulated as an energy-based model,

$$p_\alpha(y, z) = \frac{1}{Z_\alpha} \exp(\langle y, f_\alpha(z) \rangle)p_0(z),$$  \hspace{2cm} (2)

where $p_0(z)$ is a reference distribution, assumed to be isotropic Gaussian (or uniform) prior of the $d$-dimensional. Our model is defined by

$$p_\alpha(y, z) \propto \exp(\langle y, f_\alpha(z) \rangle),$$  \hspace{2cm} (3)

i.e., a softmax classifier, where $f_\alpha(z)$ provides the $K$ logit scores for the $K$ categories. Marginally,

$$p_\alpha(z) = \frac{1}{Z_\alpha} \exp(F_\alpha(z))p_0(z),$$  \hspace{2cm} (4)

where the marginal energy term

$$F_\alpha(z) = \log \sum_y \exp(\langle y, f_\alpha(z) \rangle),$$  \hspace{2cm} (5)

i.e., the so-called log-sum-exponential form. The summation can be easily computed because we only need to sum over $K$ different values of the one-hot $y$.

The above prior model $p_\alpha(y, z)$ stands on a generation model $p_\beta(x|z)$, which is assumed to be the same as the top-down network in VAE [23]. In particular, $x = g_\beta(z) + \epsilon$, where $\epsilon \sim N(0, \sigma^2 I_D)$, so that $p_\beta(x|z) \sim N(g_\beta(z), \sigma^2 I_D)$. As in VAE, $\sigma^2$ takes an assumed value.

2.2 Prior and posterior sampling: symbol-aware continuous vector computation

Sampling from the prior $p_\alpha(z)$ and the posterior $p_\theta(z|x)$ can be accomplished by Langevin dynamics. For prior sampling from $p_\alpha(z)$, Langevin dynamics iterates $z_{t+1} = z_t + s^2 \nabla_z \log p_\alpha(z_t)/2 + N(0, s^2 I_d)$, where $s$ is the step size, and the gradient is computed by

$$\nabla_z \log p_\alpha(z) \equiv \mathbb{E}_{p_\alpha(y|z)}[\nabla_z \log p_\alpha(y, z)] = \mathbb{E}_{p_\alpha(y|z)}[\langle y, \nabla_z f_\alpha(z) \rangle],$$  \hspace{2cm} (6)

where the gradient computation involves averaging $\nabla_z f_\alpha(z)$ over the softmax classification $p_\alpha(y|z)$ in [4]. Thus the sampling of the continuous dense vector $z$ is aware of the symbolic $y$.

Posterior sampling from $p_\theta(z|x)$ follows a similar scheme, where

$$\nabla_z \log p_\theta(z|x) \equiv \mathbb{E}_{p_\theta(y|z)}[\langle y, \nabla_z f_\alpha(z) \rangle] + \nabla_z \log p_\beta(x|z).$$  \hspace{2cm} (7)

When the dynamics is reasoning about $x$ by sampling the dense continuous vector $z$ from $p_\theta(z|x)$, it is aware of the symbolic $y$ via the softmax $p_\alpha(y|z)$. 

2
Thus \((y, z)\) forms a duality between symbol and dense vector, i.e., they are two sides of a single “sym-vec coin.” The entropy of \(p_\alpha(y|z)\) may determine whether the symbolic interpretation will pop out from \(z\).

Pang et al. [36] proposed to use prior and posterior sampling for maximum likelihood learning. Due to the low-dimensionality of the latent space, MCMC sampling is affordable and mixes well. Comparing prior and posterior sampling, prior sampling is particularly affordable, because \(f_\beta(z)\) is a small network. In comparison, \(\nabla z \log p_\beta(x|z)\) in the posterior sampling requires back-propagation through the generator network, which can be more expensive. Therefore we shall amortize the posterior sampling from \(p_\beta(x|z)\) by an inference network, as detailed in the next subsection.

### 2.3 Amortizing posterior sampling and variational learning

Similar to VAE [23], we recruit an inference network \(q_\phi(z|x)\) to approximate the true posterior \(p_\theta(z|x)\), in order to amortize posterior sampling. Following VAE, we learn the inference model \(q_\phi(z|x)\) and the top-down model \(p_\theta(z, x)\) jointly.

For unlabeled \(x\), the log-likelihood \(\log p_\theta(x)\) is lower bounded by the evidence lower bound (ELBO),

\[
\text{ELBO}(\theta, \phi) = \log p_\theta(x) - \mathbb{D}_{KL}(q_\phi(z|x)\|p_\theta(z|x))
\]

\[= E_{q_\phi(z|x)}[\log p_\theta(x|z)] - \mathbb{D}_{KL}(q_\phi(z|x)\|p_\alpha(z)),
\]

where \(\mathbb{D}_{KL}\) denotes the Kullback-Leibler divergence.

For the prior model, the learning gradient is

\[
\nabla_\alpha \text{ELBO} = E_{q_\phi(z|x)}[\nabla_\alpha F_\alpha(z)] - E_{p_\alpha(z)}[\nabla_\alpha F_\alpha(z)],
\]

where \(F_\alpha(z)\) is defined by (5). \(E_{q_\phi(z|x)}\) is approximated by samples from the inference network, and \(E_{p_\alpha(z)}\) is approximated by persistent MCMC samples from the prior.

Let \(\psi = \{\beta, \phi\}\) collect the parameters of the inference (encoder) and generator (decoder) models. The learning gradients for the two models are

\[
\nabla_\psi \text{ELBO} = \nabla_\psi E_{q_\phi(z|x)}[\log p_\beta(x|z)] - \nabla_\psi \mathbb{D}_{KL}(q_\phi(z|x)\|p_\theta(z)) + \nabla_\psi E_{q_\phi(z|x)}[F_\alpha(z)],
\]

where \(\mathbb{D}_{KL}(q_\phi(z|x)\|p_\theta(z))\) is tractable and the expectations in the other two terms are approximated by samples from the inference network \(q_\phi(z|x)\) with reparametrization trick [23].

VAE is employed for training latent space EBM in recent work [2, 42]. In our work, we train the inference network jointly with the latent space EBM. We only need to include the extra \(F_\alpha(z)\) term, and \(\log Z_\alpha\) is a constant that can be discarded. This expands the scope of VAE where the top-down model is a latent EBM.

As mentioned above, we shall not amortize the prior sampling from \(p_\alpha(z)\) due to its simplicity, and due to the fact that sampling \(p_\alpha(z)\) is only needed in the training stage, but is not required in the testing stage. In general, we can also amortize the sampling of \(p_\alpha(z)\) by a flow-based model [9, 10]. See appendix for a general theoretical formulation.

### 2.4 Labeled data

For a labeled example \((x, y)\), the log-likelihood can be decomposed into \(\log p_\theta(x, y) = \log p_\theta(x) + \log p_\theta(y|x)\). The gradient of \(\log p_\theta(x)\) can be computed in the same way as the unlabeled data, and

\[
p_\theta(y|x) = E_{p_\theta(z|x)}[p_\theta(y|z)] \approx E_{q_\phi(z|x)}[p_\theta(y|z)],
\]

where \(p_\theta(y|z)\) is the softmax classifier defined by (3), and \(q_\phi(z|x)\) is the learned inference network. Thus \(\nabla_\theta \log p_\theta(y|x) \approx \nabla_\theta \log E_{q_\phi(z|x)}[p_\theta(y|z)]\). The gradients backpropagate to both the EBM prior and the inference network.

For semi-supervised learning, we can combine the learning gradients from both unlabeled and labeled data.
2.5 Algorithm

The learning and sampling algorithm is described in Algorithm 1.

**Algorithm 1**: Semi-supervised learning of latent space EBM with symbol-vector coupling.

- **input**: Learning iterations \(T\), learning rates \((\eta_0, \eta_1, \eta_2)\), initial parameters \((\alpha_0, \beta_0, \phi_0)\), observed unlabelled examples \(\{x_i\}_{i=1}^M\), observed labelled examples \(\{(x_i, y_i)\}_{i=M+1}^{M+N}\), unlabelled and labelled batch sizes \((\alpha, \beta)\), set of persistent chains \(\{z_i^+ \sim p_0(z)\}_{i=1}^{L}\), and number of Langevin dynamics steps \(T_{LD}\).

- **output**: \((\alpha_T, \beta_T, \phi_T)\).

for \(t = 0 : T - 1\) do

1. **Mini-batch**: Sample unlabelled \(\{x_i\}_{i=1}^m\) and labelled observed examples \(\{x_i, y_i\}_{i=m+1}^{m+n}\).

2. **Prior sampling**: For each unlabelled \(x_i\), randomly pick and update a persistent chain \(z_i^-\) by MCMC with target distribution \(p_0(z)\) for \(T_{LD}\) steps.

3. **Posterior sampling**: For each \(x_i\), sample \(z_i^+ \sim q_\phi(z|x_i)\) using the inference network and reparameterization trick.

4. **Unsupervised learning of prior model**: \(\alpha_{t+1} = \alpha_t + \eta_0 \frac{1}{m} \sum_{i=1}^m [\nabla_\alpha F_{\alpha_t}(z_i^+) - \nabla_\alpha F_{\alpha_t}(z_i^-)]\).

5. **Unsupervised learning of inference and generator models**: \(\psi_{t+1} = \psi_t + \eta_1 \frac{1}{m} \sum_{i=1}^m [\nabla_\psi \log p_{\beta_t}(z|x_i)] - \nabla_\psi \mathbb{D}_{KL}(q_{\phi_t}(z|x_i) || p_0(z)) + \nabla_\psi [F_{\alpha_t}(z_i^+)]\).

6. **Supervised learning of prior and inference model**: \(\theta_{t+1} = \theta_t + \eta_2 \frac{1}{m+n} \sum_{i=m+1}^{m+n} \sum_{k=1}^K y_{i,k} \log(p_{\theta_t}(y_{i,k}|z_i^+))\).

3 Related work

**Discriminative models.** Recent semi-supervised learning (SSL) methods based on discriminative models have achieved substantial progress. Most of these methods rely on data augmentation strategies which heavily exploit class-invariance properties of images [3, 40]. Invariance properties for text and tabular data are however less clear. One successful method, virtual adversarial training (VAT) [31], does not depend on image-domain-specific properties. Instead, it finds an adversarial augmentation to \(x\) within an \(\epsilon\)-ball of \(x\) with respect to \(l_p\) norm such that the distance between the class distribution conditional on \(x\) and that conditional on the augmentation is maximized. This approach is applicable to continuous data beyond images like tabular data. Nevertheless still has the limitation on discrete data since the adversarial augmentation requires the data space to be differentiable.

**GANs.** Generative models in the GAN family have also been applied to SSL and exhibited promising results. Salimans et al. [39] changed the discriminator in GAN to be a classifier with an extra “generated” class, which is thus able to simultaneously distinguish real and generated samples and to predict class labels. To avoid the incompatible roles of the classifier and discriminator, Li et al. [5] proposed to keep the discriminator in the original GAN and add a classifier for class prediction. Dai et al. [7] argued that a preferred generator for SSL should be a complement generator in the sense that it should assign high densities for data points with low densities in the data distribution. These GAN-based SSL methods generally perform well on images, competitive with discriminative models. However, GANs are highly tuned to image data. The adversarial generative training relies on the differentiability of data generation and applying it to discrete data like text with does not exhibit much gain if no harm is caused [4].

**Energy-based model.** Recently EBMs have also been applied to SSL. Grathwohl et al. [14] proposed Joint Energy-based Model (JEM) on top of a regular discriminative classifier by re-interpreting the logits. Zhao et al. [46] extended JEM to SSL and obtained reasonable performance. With a similar formulation, Gao et al. [11] adversarially trained the EBM with a pretrained GLOW [21] and obtained performance competitive with GAN-based and discriminative models, especially after it is combined with VAT. Our model shares similarities with these models in that our model has a latent space EBM and it is constructed through re-using the logits of a discriminative classifier in the latent space. Our EBM in the latent space however has a much lower dimension and smoother landscape, which renders fast sampling and mixing. The learning also does not require expensive pretraining of GLOW like in
Also, it is not easy to apply data space EBM to discrete data since its training requires either the data space or the generation process of the auxiliary generator (e.g., GLOW) to be differentiable.

Likelihood-based models. In a seminal work [22], Kingma et al. adopted VAE for SSL where a discrete latent variable, $y$, representing the class labels is introduced in addition to the usual continuous vector, $z$. In [30], adversarial training was applied to force $y$ and $z$ to capture disentangled semantics. In our model, the continuous vector is coupled with the discrete variable or symbol to form a latent space EBM prior. More recently, Izmailov et al. [18] proposed a flow-based model, FlowGMM, where the data distribution is mapped to a Gaussian mixture in the latent space with invertible flow. Our model is similar in the sense that it uses an inference network to map the data distribution to an latent space EBM. But our model does not have the restriction of invertibility. FlowGMM is aimed for broad applicability and demonstrate promising results on text and tabular data, sharing the goal of our work.

4 Experiments

We evaluate our model on a wide variety of datasets to demonstrate the broad applicability of our model. We begin the evaluation with image data which most SSL research works have been focused on. Next we consider text data in both continuous and discrete situations. Finally, our model is assessed on tabular data. Two recent works applied their methods, FlowGMM [18] and JEM [46], beyond image data and these two models are closely related to ours. They thus serve as important baselines of our model.

For image experiments, the inference network is implemented with a standard Wide ResNet [44], widely used in image SSL works [35]. The generator network is a 4-layer deconvolutional network, similar to the generator in DCGAN [38]. The inference and generator networks in text and tabular experiments are all a 3-layer MLP. The latent EBM prior in all experiments is also a small 3-layer MLP with a hidden dimension of 200. All model parameters are initialized with Xavier normal [12]. Adam [20] is adopted for all model optimization. The models are trained until convergence (taking approximately 400,000 parameter updates for image models, 100,000 updates for text models without pretrained embeddings, 3,000 updates for text models with pretrained embeddings, 4,000 updates for tabular models). We run 20 steps of persistent chains with step size 0.6 to obtain MCMC samples used in learning the latent space EBM prior (i.e., Step 4 in Algorithm 1). Notice that MCMC sampling only needs to compute the gradients of the small latent space EBM and does not need to backpropagate through the larger generator.

4.1 Image

Standard image SSL benchmarks, SVHN [32], with 1,000 labeled data and 64,932 unlabeled data, and CIFAR-10 [24], with 4,000 labeled data and 41,000 unlabeled data, are adopted to assess our model on image classification. The results are summarized in Table 1. Besides FlowGMM [18] and JEM [46], we also compare our model to other likelihood-based models, VAE M1+M2 [22], AAE [30], in addition to representative GAN-based models, TripleGAN [5], BadGAN [7] and discriminative models, II-Model [26] and VAI [31]. Our model outperforms FlowGMM and JEM and other likelihood-based models. The improvement is especially clear on SVHN (with almost 10% absolute improvement compared to FlowGMM). We however observe a performance gap between our model and GAN-based and discriminative methods which highly tuned for images.

4.2 Text

We evaluate our model on text with a widely used text classification dataset, AGNews [45], in two settings. The first one follows the evaluation protocol used in FlowGMM [18]. Text data are first embedded by a pretrained language model, BERT [8], and the SSL model is constructed on the embedding space. We use the preprocesed data provided by [18] which contains 200 labeled data and 120,000 unlabeled data are used in training. AGNews has 4 classes. Our model is compared to FlowGMM [18], RBF Label Spreading (a graph-based label spreading method) [47], and II-Model [26]. Table 2 displays the results. Our model exhibits competitive performance with FlowGMM and outperforms other baselines.
Table 1: Accuracy on SVHN and CIFAR-10.

<table>
<thead>
<tr>
<th>Method</th>
<th>SVHN 1000 Labels</th>
<th>CIFAR-10 4000 Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>VAE M1+M2</td>
<td>64.0 ± 0.1</td>
<td>-</td>
</tr>
<tr>
<td>AAE</td>
<td>82.3 ± 0.3</td>
<td>-</td>
</tr>
<tr>
<td>JEM</td>
<td>66.0 ± 0.7</td>
<td>-</td>
</tr>
<tr>
<td>FlowGMM</td>
<td>82.4</td>
<td>78.2</td>
</tr>
<tr>
<td>Ours</td>
<td>92.0 ± 0.1</td>
<td>78.6 ± 0.3</td>
</tr>
<tr>
<td>TripleGAN</td>
<td>94.2 ± 0.2</td>
<td>83.0 ± 0.4</td>
</tr>
<tr>
<td>BadGAN</td>
<td>95.8 ± 0.03</td>
<td>85.6 ± 0.03</td>
</tr>
<tr>
<td>Π-Model</td>
<td>94.6 ± 0.2</td>
<td>83.6 ± 0.3</td>
</tr>
<tr>
<td>VAT</td>
<td>96.3 ± 0.1</td>
<td>88.0 ± 0.1</td>
</tr>
</tbody>
</table>

Table 2: Accuracy on AGNews with Bert embeddings.

<table>
<thead>
<tr>
<th>Method</th>
<th>AGNews-Bert 200 Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBF Label Spreading</td>
<td>36.1</td>
</tr>
<tr>
<td>FlowGMM</td>
<td>82.1 ± 1.0</td>
</tr>
<tr>
<td>Ours</td>
<td>82.0 ± 0.2</td>
</tr>
<tr>
<td>Π-Model</td>
<td>80.2 ± 0.3</td>
</tr>
</tbody>
</table>

Table 3: Accuracy on AGNews with Unigram.

4.3 Tabular data

We use three tabular datasets from the UCI repository. Hepmass and Miniboone were utilized in [18] for SSL and Protein was used in [46]. Protein has a continuous target variable and we follow [46] to bin the targets into 10 equally weighted buckets. We use the same experimental settings as in the two prior works. The number of labeled / unlabeled data for Hepmass, Miniboone, and Protein are 20/140,000, 20/65,000, 100/41,057 respectively. Hepmass and Miniboone have 2 classes, while Protein has 10 classes. We compare our model to RBF Label Spreading, JEM, FlowGMM, Π-Model,
and VAT. Table 4 summarizes the results. Our model outperforms all baselines across the three tabular datasets.

<table>
<thead>
<tr>
<th>Method</th>
<th>Hepmass 20 Labels</th>
<th>Miniboone 20 Labels</th>
<th>Protein 100 Labels</th>
</tr>
</thead>
<tbody>
<tr>
<td>RBF Label Spreading</td>
<td>84.9</td>
<td>79.3</td>
<td>-</td>
</tr>
<tr>
<td>JEM</td>
<td>-</td>
<td>-</td>
<td>19.6</td>
</tr>
<tr>
<td>FlowGMM</td>
<td>88.5 ± 0.2</td>
<td>80.5 ± 0.7</td>
<td>-</td>
</tr>
<tr>
<td>Ours</td>
<td>89.1 ± 0.1</td>
<td>81.2 ± 0.3</td>
<td>23.1 ± 0.3</td>
</tr>
<tr>
<td>Π-Model</td>
<td>87.9 ± 0.2</td>
<td>80.8 ± 0.01</td>
<td>-</td>
</tr>
<tr>
<td>VAT</td>
<td>-</td>
<td>-</td>
<td>17.1</td>
</tr>
</tbody>
</table>

Table 4: Accuracy on Hepmass, Miniboone, and Protein.

5 Conclusion

Semi-supervised learning based on latent space EBM prior with symbol-vector coupling is very natural. For unlabeled data, the marginal EBM prior is in the form of sum of exponentials. For labeled data, the conditional distribution of label given the inferred latent vector is a regular softmax classifier. The semi-supervised learning can be based on a principled likelihood-based framework, with inference computation being amortized by a variational inference network.

Our model may be interpreted as a generative classifier, where the latent vector used for classification is inferred based on a top-down generative model. The top-down model and the posterior inference captures the concept of information bottleneck [41] more naturally than bottom-up classifier. The posterior inference of a top-down model may be more robust to adversarial perturbations than a classifier defined on the input directly, because the posterior inference can explain away the adversarial perturbations via the top-down model. The inference of the latent vector is aware of the underlying symbol, and the symbol-vector coupling in our prior model may shed light on the interaction between symbolic reasoning and continuous computation. We may consider a multi-layer top-down model where each layer consists of dense sub-vectors coupled with symbolic one-hot sub-vectors, so that continuous computation based on dense sub-vectors is aware of the corresponding symbols.

Our experiments show that our semi-supervised learning method outperforms existing methods on text and tabular data. We shall continue to improve our method on image data. To rephrase the title of this workshop, we cannot believe our principled model-based semi-supervised learning method does not work better than existing methods on image data.

Acknowledgement

We thank the reviewers for their insightful comments and suggestions. The work is supported by NSF DMS-2015577.

Appendix: learning latent EBM with divergence perturbation formulation

This section provides a theoretical formulation for learning latent EBM in general, where our model is a special case. We can embed our learning method within this general formulation.

The latent EBM is of the following form:

\[ p_\theta(z, x) = \frac{1}{Z_\theta} \exp(f_\theta(x, z)). \]  (13)

Marginally, \( p_\theta(x) = \int p_\theta(z, x) dz \), and the posterior is \( p_\theta(z|x) = p_\theta(z, x)/p_\theta(x) \).

Let \( p_{\text{data}}(x) \) be the data distribution that generates \( x \). Maximum likelihood estimation (MLE) minimizes \( D_{\text{K-L}}(p_{\text{data}}(x)||p_\theta(x)) \), where expectation with respect to \( p_{\text{data}} \) can be approximated by averaging over observed examples.
Learning of latent EBM can be based on the following divergence perturbation (see [15] [16] for related formulations).

\[ \Delta = D_{\text{KL}}(p_{\text{data}}(x)||p_\theta(x)) + D_{\text{KL}}(q_+(z|x)||p_\theta(z|x)) - D_{\text{KL}}(q_-(z,x)||p_\theta(z,x)), \]  

(14)

which is a perturbation of the divergence \( D_{\text{KL}}(p_{\text{data}}(x)||p_\theta(x)) \) that underlies MLE. \( q_+(z|x) \) is the positive phase sampler, and \( q_-(z,x) \) is the negative phase sampler. They correspond to the positive phase and negative phase of MLE learning of latent EBM such as Boltzmann machine [1]. For two distributions \( q(z|x) \) and \( p(z|x) \), we define \( D_{\text{KL}}(q(z|x)||p(z|x)) = \mathbb{E}_{p_{\text{data}}(x)}[\mathbb{E}_{q(z|x)}[\log q(z|x)/p(z|x)]] \), with an outer expectation with respect to \( p_{\text{data}} \). The divergence \( D_{\text{KL}}(p_{\text{data}}(x)||p_\theta(x)) \) for MLE is perturbed by the positive phase divergence \( D_{\text{KL}}(q_+(z|x)||p_\theta(z|x)) \) and the negative phase divergence \( D_{\text{KL}}(q_-(z,x)||p_\theta(z,x)) \).

For latent space EBM prior, we have the factorization \( p_\theta(z,x) = p_\alpha(z)p_\beta(x|z) \), and \( q_-(z,x) = q_-(z)p_\beta(x|z) \), so that the negative phase \( D_{\text{KL}} \) in (14) becomes \( D_{\text{KL}}(q_-(z)||p_\alpha(z)) \), i.e., the KL divergence between the priors.

The above formulation has the following features.

1. Gradient of divergence perturbation \( \Delta \) can be calculated explicitly without intractable integrations in \( p_\theta(x) \) and log \( Z_\theta \), because \( p_\theta(x) \) is merged with \( p_\theta(z|x) \) to become \( p_\theta(z,x) \) due to the positive phase \( D_{\text{KL}} \) term, and the log \( Z_\theta \) is canceled by the negative phase \( D_{\text{KL}} \) term.

2. For MLE learning, at iteration \( t \), let \( \theta_t \) be the current estimate of \( \theta \). We let \( q_+(z|x) = p_{\theta_t}(z|x) \) and \( q_-(z,x) = p_{\theta_t}(z,x) \). In this case, the positive phase \( D_{\text{KL}}(q_+(z|x)||p_\theta(z|x)) \) achieves its minimum 0 at \( \theta_t \), so that its gradient at \( \theta_t \) is 0. Similarly the negative phase \( D_{\text{KL}}(q_-(z,x)||p_\theta(z,x)) \) achieves its minimum 0 at \( \theta_t \) too, so that its gradient at \( \theta_t \) is also 0. Thus the gradient of the divergence perturbation \( \Delta \) in (14) is the same as the MLE gradient of the first KL term \( D_{\text{KL}}(p_{\text{data}}(x)||p_\theta(x)) \).

3. For amortized computation, \( q_+(z|x) = q_{\phi_+}(z|x) \) is an inference network, and \( q_-(z,x) = q_{\phi_-}(z,x) \) is a synthesis network, with parameters \( \phi_+ \) and \( \phi_- \) respectively, that are independent of the model parameter \( \theta \). The learning can be based on \( \min_{\theta} \min_{\phi_+} \max_{\phi_-} \Delta \). The positive phase \( D_{\text{KL}} \) leads to variational learning. The negative phase \( D_{\text{KL}} \) leads to a generalized form of contrastive divergence [17], as well as adversarial interpretation due to the negative sign and \( \max_{\phi_-} \).

For latent space EBM prior, we only need a synthesis network \( q_{\phi_-}(z) \) to approximate the prior model \( p_\alpha(z) \). We may consider using a flow-based model [9] [10] for \( q_{\phi_-}(z) \).

In our current work, we employ an inference network for \( q_{\phi_+}(z|x) \), but we leave \( q_{\phi_-}(z) \) to MCMC sampling, because \( p_\alpha(z) \) is very simple, and sampling of \( p_\alpha(z) \) does not require back-propagation through the generator network. Moreover, sampling \( p_\alpha(z) \) is not needed in the testing stage.

4. In between (2) and (3), both \( q_+(z|x) \) and \( q_-(z,x) \) can be sampled by short-run MCMC for inference and synthesis [15] [24]. The short-run MCMC runs a fixed number of MCMC iterations from a fixed initial distribution, so that both \( q_+(z|x) \) and \( q_-(z,x) \) are well defined. The gradient of \( \Delta \) in (14) based on short-run MCMC is a perturbation of the MLE learning gradient. The MLE learning gradient is impractical because it requires convergence of MCMC. Learning algorithm based on short-run MCMC converges to the solution of an estimating equation which is a perturbation of the MLE estimating equation. The learning algorithm of Pang et al. [36] works within this scheme.

The above formulation encompasses many generative models and associated learning algorithms. It helps us understand the model and learning algorithm proposed in this paper. It will also help us develop new models and algorithms.

References


