Variational Inference In Pachinko Allocation Machines

Akash Srivastava
University of Edinburgh
akash.srivastava@ed.ac.uk

Charles Sutton
University of Edinburgh
The Alan Turing Institute
Google Brain
csutton@inf.ed.ac.uk

Abstract

The Pachinko Allocation Machine (PAM) is a deep topic model that allows representing rich correlation structures among topics by a directed acyclic graph over topics. Because of the flexibility of the model, however, approximate inference is very difficult. Perhaps for this reason, only a small number of potential PAM architectures have been explored in the literature. In this paper we present an efficient and flexible amortized variational inference method for PAM, using a deep inference network to parameterize the approximate posterior distribution in a manner similar to the variational autoencoder. Our inference method produces more coherent topics than state-of-art inference methods for PAM while being an order of magnitude faster, which allows exploration of a wider range of PAM architectures than have previously been studied.

1 Introduction

Topic models are widely used tools for exploring and visualizing document collections. Simpler topic models, like latent Dirichlet allocation (LDA) (Blei et al., 2003), capture correlations among words but do not capture correlations among topics. This limits the model's ability to discover finer-grained hierarchical latent structure in the data. For example, we expect that very specific topics, such as those pertaining to individual sports teams, are likely to co-occur more often than more general topics, such as a generic "politics" topic with a generic "sports" topic.

A popular extension to LDA that captures topic correlations is the Pachinko Allocation Machine (PAM) (Li and McCallum, 2006). PAM is essentially "deep LDA". It is defined by a directed acyclic graph (DAG) in which each leaf node denotes a word in the vocabulary, and each internal node is associated with a distribution over its children. The document is generated by sampling, for each word, a path from the root of the DAG to a leaf. Thus the internal nodes can represent distributions over topics, so-called "super-topics", and so on, thereby representing correlations among topics.

Unfortunately PAM introduces many latent variables — for each word in the document, the path in the DAG that generated the word is latent. Therefore, traditional inference method, such as Gibbs sampling and decoupled mean-field variational inference become significantly more expensive. This not only affects the scale of data sets that can be considered, but more fundamentally the computational cost of inference makes it difficult to explore the space of possible architectures for PAM. As a result, to date only relatively simple architectures have been studied in the literature (Li and McCallum, 2006; Mimno et al., 2007; Li et al., 2012).

We present what is, to the best of our knowledge, the first variational inference method for PAM, which we call dnPAM. Unlike collapsed Gibbs, dnPAM can be generically applied to any PAM architecture without the need to derive a new inference algorithm, allowing much more rapid exploration of the space of possible model architectures. dnPAM is an *amortized inference* following the learning principle of variational autoencoders, which means that the variational distribution is parameterized by a deep network, which is trained to perform accurate inference. We find that dnPAM is not only an order of magnitude faster than collapsed Gibbs, but even returns topics with comparable or greater coherence. The dramatic speedup in inference time comes from the amortization of the learning cost via learning a neural network to produce posterior parameter instead of learning these parameters directly. This efficiency in inference enables exploration of more complex and deeper PAM models than have previously been possible.

As a demonstration of this, as our second contribution we introduce a mixture of PAM model, where each component distribution of the mixture is represented by a PAM. By mixing PAMs with varying number of topics, this model captures the latent structure in the data at many different levels of granularity that decouples general broad topics from the more specific ones.

Like other variational autoencoders (VAEs) (Kingma and Welling, 2013; Rezende et al., 2014), our model also suffers from the posterior collapsing (van den Oord et al., 2017) what is sometime also referred to as component collapsing (Dinh and Dumoulin, 2016) and slow training due to low learning rates. We present an analysis of these issues in the context of topic modeling and propose normalization based solution to alleviate them.

2 Latent Dirichlet Allocation

LDA represents each document $\mathbf{w}$ in a collection as a mixture of topics. Each topic vector β_k is a distribution over the vocabulary, that is, a vector of probabilities, and $\beta = (\beta_1 \dots \beta_K)$ is the matrix of the K topics. Every document is then modeled as an admixture of the topics. The generative process is to first sample a proportion vector $\theta \sim \text{Dirichlet}(\alpha)$, and then for each word at position n, sampling a topic indicator $z_n \in \{1, \dots K\}$ as $z_n \sim \text{Categorical}(\theta)$, and finally sampling the word index $w_n \sim \text{Categorical}(\beta_{z_n})$.

2.1 Deep LDA: Pachinko Allocation Machine

PAM is a class of topic models that extends LDA by modeling correlations among topics. A particular instance of a PAM represents the correlation structure among topics by a DAG in which the leaf nodes represent words in the vocabulary and the internal nodes represent topics. Each node s in the DAG is associated with a distribution θ_s over its children, which has a Dirichlet prior. There is no need to differentiate between nodes in the graph and the distributions θ_s, so we will simply take $\{\theta_s\}$ to be the node set of the graph. To generate a document in PAM, for each word we sample a path from the root to a leaf, and output the word associated with that leaf.

More formally, we present the special case of 4-PAM, in which the DAG is a 4-partite graph. It will be clear how to generalize this discussion to arbitrary DAGs. In 4-PAM, the DAG consists of a root node θ_r which is connected to children $\theta_1 \dots \theta_S$ called *super-topics*. Each super-topic θ_s is connected to the same set of children $\beta_1 \dots \beta_K$ called subtopics, each of which are fully connected to the vocabulary items $1 \dots V$ in the leaves.

A document is generated in 4-PAM as follows. First, a single matrix of subtopics β are generated for the entire corpus as $\beta_k \sim \text{Dirichlet}(\alpha_0)$. Then, to sample a document $\mathbf{w}$, we sample child distributions for each remaining internal node in the DAG. For the root node, θ_r is drawn from a Dirichlet prior $\theta_r \sim \text{Dirichlet}(\alpha_r)$, and similarly for each super-topic $s \in \{1 \dots S\}$, the supertopic θ_s is drawn as $\theta_s \sim \text{Dirichlet}(\alpha_s)$. Finally, for each word w_n, a path is sampled from the root to the leaf. From the root, we sample the index of a supertopic $z_{n0} \in \{1 \dots S\}$ as $z_{n0} \sim \text{Categorical}(\theta_r)$, followed by a subtopic index $z_{n2} \in \{1 \dots K\}$ sampled as $z_{n2} \sim \text{Categorical}(\beta_{z_{n1}})$, and finally the word is sampled as $w_n \sim \text{Categorical}(\beta_{z_{n1}})$. This process can be written as a density

$$P(\mathbf{w}, \mathbf{z}, \theta \mid \boldsymbol{\alpha}, \boldsymbol{\beta}) = p(\theta_r|\alpha_r) \prod_{s=1}^{S} P(\theta_s|\alpha_s) \tag{1}$$
$$\times \prod_{n} p(z_{n1}|\theta_r) p(z_{n2}|\theta_{z_{n1}}) p(w_n|\beta_{z_{n2}}).$$

It should be easily seen how this process can be extended to arbitrary ℓ-partite graphs, yielding the ℓ-PAM model, in which case LDA exactly corresponds to 3-PAM, and also to arbitrary DAGs.

3 Mixture of PAMs

The main advantage of the inference framework we propose is that it allows easily exploring the design space of possible structures for PAM. As a demonstration of this, we present a word-level mixture of PAMs that allows learning finer grained topics than a single PAM, as some mixture components learn topics that capture the more general, global topics so that other mixture components can focus on finer-grained topics.

We describe a word-level mixture of M PAMs $P_1 \dots P_M$, each of which can have a different number of topics or even a completely different DAG structure. To generate a document under this model, first we sample an M-dimensional document level mixing proportion $\theta_r \sim \text{Dirichlet}(\alpha_r)$. Then, for each word w_n in the document, we

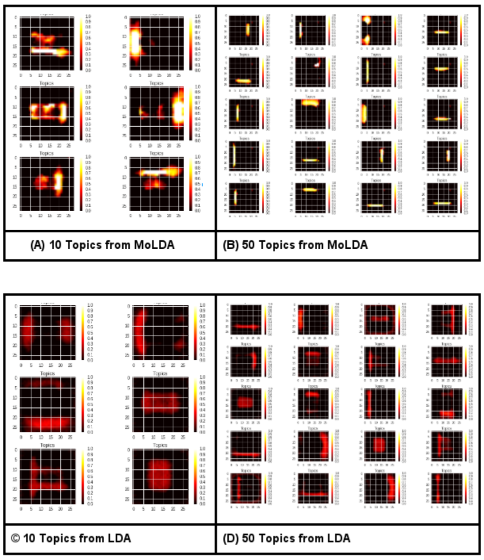

Figure 1: Top: A and B show randomly sampled topics from MoLDA(10:50). Bottom: C and D show randomly sampled topics from LDA with 10 topics and 50 topics on Omniglot. Notice that by using a mixture, the MoLDA can decouple the higher level structure (A) from the lower-level details(B).

choose one of the PAM models by sampling $m \sim \text{Categorical}(\theta_r)$ and then finally sample a word by sampling a path through P_m as described in the previous section. This model can be expressed as a general PAM model in which the root node θ_r is connected to the root nodes of each of the M mixture components. If each of the mixture components are 3-PAM models, that is LDA, then we call the resulting model a mixture of LDA models (MoLDA).[1]

The advantage of this model is that if we choose to incorporate different mixture components with different numbers of topics, we find that the components with fewer topics explain the coarse-grained structure in the data, freeing up the other components to learn finer grained topics. For example, the Omniglot dataset contains 28x28 images of handwritten alphabets from artificial scripts. In Figure 1, panels (C) and (D) are visualization of the latent topics that are generated using vanilla LDA with 10 and 50 topics, respectively. Because we are modelling image data, each topic can also be visualized as an image. Panels (A) and (B) show the topics from a single MoLDA with two components, one with 10 topics and one with 50 topics. It is apparent that the MoLDA topics are sharper, indicating that each individual topic is capturing more information about the data. The mixture model allows the two LDAs being mixed to focus exclusively on higher (for 10 topics) and lower (for 50 topics) level features while modeling the images. On the other hand, the topics in the vanilla LDA need to account for all the variability in the dataset using just 10 (or 50) topics and therefore are fuzzier.

[1]It would perhaps be more proper to call this model an *admixture* of LDA models.

4 Inference

Probabilistic inference in topic models is the task of computing posterior distributions $p(\mathbf{z}|\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ over the topic assignments for words, or over the posterior $p(\theta|\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ of topic proportions for documents. For all practical topic models, this task is intractable. Commonly used methods include Gibbs sampling (Li and McCallum, 2006; Blei et al., 2004), which can be slow to converge, and variational inference methods such as mean field (Blei et al., 2003; Blei and Lafferty, 2006), which sometimes sacrifice topic quality for computational efficiency. More fundamentally, these families of approximate inference algorithms tend to be model specific and require extensive mathematical sophistication on the practitioner's part since even the slightest changes in model assumptions may require substantial adjustments to the inference. The time required to derive new approximate inference algorithms dramatically slows explorations through the space of possible models.

In this work we present a generic, amortized approximate inference method dnPAM for learning in the PAM family of models, that is extremely fast, flexible and accurate. The inference method is flexible in the sense that it can be generically applied to any DAG structure for PAM, without the need to derive a new variational update. The main idea is that we will approximate the posterior distribution $p(\theta_s|\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$ for each super-topic θ_s by a *variational distribution* $q(\theta_s|\mathbf{w})$. Unlike standard mean field approaches, in which $q(\theta_s|\mathbf{w})$ has an independent set of variational parameters for each document in the corpus, the parameters of $q(\theta_s|\mathbf{w})$ will be computed by an *inference network*, which is a neural network that takes the document $\mathbf{w}$ as input, and outputs the parameters of the variational distribution. This is motivated by the observation that similar documents can be described well by similar posterior parameters.

In dnPAM, we seek to approximate the posterior distribution $P(\theta|\mathbf{w}, \boldsymbol{\alpha}, \boldsymbol{\beta})$, that is, the paths z_n for each word are integrated out. Note that this is in contrast to previous collapsed Gibbs methods for PAM (Li and McCallum, 2006), which integrate out θ using conjugacy. To simplify notation, we will describe dnPAM for the special case of 4-PAM, but it will be clear how to generalize this discussion to arbitrary DAGs. So for 4-PAM, we have $\theta = (\theta_r, \theta_1 \ldots \theta_S)$. We introduce a variational distribution $q(\theta|\mathbf{w}) = q(\theta_r|\mathbf{w})q(\theta_1|\mathbf{w}) \ldots q(\theta_S|\mathbf{w})$.

To choose the best approximation $q(\theta|\mathbf{w})$, we construct a lower bound to the evidence (ELBO) using Jensen's inequality, as is standard in variational inference. For example, the log-likelihood function $\log p(\mathbf{w}|\alpha, \beta)$ for the 4-PAM model (1) can be lower bounded by

$$\begin{aligned} \mathcal{L} = &- \mathrm{KL}[q(\theta_r|\mathbf{w})||p(\theta_r|\alpha_r)] \\ &- \sum_{s=1}^{S} \mathrm{KL}[q(\theta_s|\mathbf{w})||p(\theta_s|\alpha_s)] \\ &+ \mathbb{E}\left[\sum_n \log p(w_n|\theta, \beta)\right], \end{aligned} \tag{2}$$

where the expectation is with respect to the variational posterior $q(\theta|\mathbf{w})$.

dnPAM uses stochastic gradient descent to maximize this ELBO to infer the variational parameters and learn the model parameters. To finish describing the method, we must describe how q is parameterized, which we do next. For the subtopic parameters β, we learn these using variational EM, that is, we maximize $\mathcal{L}$ with respect to β. It would be a simple extension to add a variational distribution over β if this was desired.

Re-parameterizing Dirichlet Distribution: The expectation over the second term in equation (2) is in general intractable and therefore we approximate it using a special type of Monte-Carlo (MC) method (Kingma and Welling, 2013; Rezende and Mohamed, 2015) that employs the re-parametrization-trick (Williams, 1992) for sampling from the variational posterior. But this MC-estimate requires $q(\theta|\mathbf{w})$ to belong to the location-scale family which excludes Dirichlet distribution. Recently, some progress has been made in the re-parametrization of distributions like Dirichlet (Ruiz et al., 2016) but in this work, following Srivastava and Sutton (2017) we approximate the posterior with a logistic normal distribution. First, we construct a Laplace approximation of the Dirichlet prior in the softmax basis, which allows us to approximate the posterior distribution using a Gaussian that is in the location-scale family. Then in order to sample θ's from the posterior in the simplex basis we apply the softmax transform to the Gaussian samples. Using this Laplace approximation trick also allows handling different prior assumptions, including other non-location-scale family distributions.

Amortizing Super-Topics: As mentioned above, in PAM the super topics need to be sampled for each document in the corpus. This presents a bottleneck in speeding up posterior inference via Gibbs sampling or DMFVI as the number of parameters to be learned increase drastically with data compared to a typical LDA. We design our inference method to tackle this bottleneck such that the number of posterior parameters to be learned do not directly depend on the number of documents in the corpus.

dnPAM Inference Network Recently Srivastava and Sutton (2017) amortized the cost of learning posterior parameter in LDA by using a feedforward Multi-layer Perceptron (MLP) to generate the parameters for the posterior distribution over the topic proportion vector θ. Like them, we model the posterior $q(\theta_r|\mathbf{w})$ as $\mathrm{LN}(\theta; f_\mu(\mathbf{w}), f_u(\mathbf{w}))$ where f_μ and f_u are neural networks that generate the parameters for the logistic normal distribution. But, since their *inference network* does not allow sampling topics, therefore they assumed the topics to be fixed model parameters. We now introduce an alternative inference network architecture that is designed to efficiently sample all the posterior parameters that need to be inferred in PAMs including topics.

For generality we assume that at the sub-topics at the lowest level are sampled only once for the corpus. To generate the parameters of the variational posterior distributions at each of the levels above, we use one MLP per level. For example in the case of 4-PAM, the parameters of the variational posteriors $(q(\theta_1|\mathbf{w}), ..., q(\theta_s|\mathbf{w}))$ over the super-topics are generated from a single MLP. These MLPs are trained using the VAE-based variational learning principle for topic models (Srivastava and Sutton, 2017) and then sampled from using the process for Dirichlet distribution described above to generate θ's and the super-topics.

Note that the Dirichlet distribution is a conjugate prior to the multinomial distribution. This fact can be used to leverage the modern GPU-based computation to generate the posterior parameters for the nodes in the next lower level since it only involves a dot-product. Therefore we stack all the topic vectors (each sampled from its respective variational posterior) in a 3-D tensor and using a custom implementation for this dot-product [2] we gain significant reduction in training time. We want to point out that the result of above process can also be seen as construction of MLPs on the fly by sampling Dirichlet vectors from our inference networks and stacking them to form weight matrices of the MLPs.

The decoder in the case of PAM is just a dot product between the sample from the output distribution of the inference network θ and the sub-topic matrix β. This makes the entire class of PAM-type mixed membership models permeable to deep learning while being Bayesian about the latent beliefs. Though in our experiments we always use MLPs to encode the posterior and decode the output, if required other architectures like CNNs and RNNs can be easily used to replace the MLPs. As mentioned before, dnPAM can work with non-Dirichlet priors by using the Laplace approximation trick. It can also handle full-covariance Gaussian as well as logistic Normals by simply using the Cholesky decomposition and can therefore be used to learn Correlated Topic Model (CTM) (Blei and Lafferty, 2006).

At first, the use of an inference network seems strange, as coupling the variational parameters across documents guarantees that the variational bound will not be as tight. But the advantage of an inference network is that after the weights of the inference network have been learned on training documents, we can obtain an approximate posterior distribution for a new test document simply by evaluating the inference network, without needing to carry out any variational optimization. This is the reason for the term *amortized inference*, i.e., the computational cost of training the inference network is amortized across future test documents.

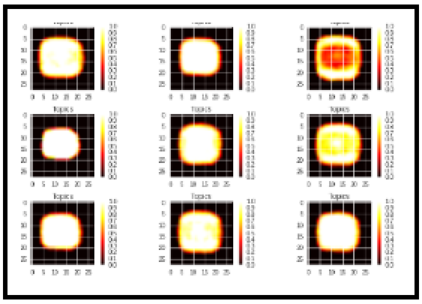

Figure 2: 9-randomly sampled "topics" from Omniglot dataset folded back to the original image dimensions. An example of how the topics look like if component collapsing occurs.

4.1 Learning Issues in VAE

Trained with stochastic variational inference, like VAEs, our PAM models suffer from primarily two learning problems; slow convergence and component collapse.

Slow Learning

Training PAM models even on the recommended learning rate of 0.001 for the ADAM optimizer (Kingma and Ba, 2014), generally causes the gradients to diverge early on in training. Therefore in practice, fairly low learning rates have been used in VAE-based generative models of text, which significantly delays the learning in such model. In this section we first explain one of the reasons for the diverging behavior of the gradients and then propose a solution that stabilizes them and therefore allows training VAEs on high learning rates, hence speeds-up the learning.

Consider a VAE for a model $p(x, z)$ where z is a latent Gaussian variable, x is a categorical variable distributed as $p_{\Theta_d}(x|z) =$ Multinomial$(f_d(z, \Theta_d))$, and the function $f_d()$ is a decoder MLP with parameters Θ_d whose outputs lie in the unit simplex. Suppose we define a variational distribution $q_{\Theta_e}(z|x) = \mathcal{N}(\mu, \exp(u))$, where $\mu = f_\mu(x, \Theta_\mu)$, $u = f_u(x, \Theta_u)$ are MLPs with parameters $\Theta_e = \{\Theta_\mu, \Theta_u\}$ and u is the logarithm of the diagonal of the covariance matrix.

Now the VAE objective function is

$$\text{ELBO}(\mathbf{\Theta}) = -KL[q_{\Theta_e}(z|x)||p(z)] + \mathbb{E}[\log p_{\Theta_d}(x|z)]. \quad (3)$$

Notice that the first term, the KL divergence, interacts only with the encoder parameters. The gra-

[2] Tensorflow requires that the rank of the matrices in tf.matmul be the same.

dients of this term $L = KL[q_{\Theta_e}(z|x)||p(z)]$ with respect to u is

$$\nabla_u L = \frac{1}{2}(\exp(u) - 1). \quad (4)$$

One explanation for the diverging behavior of the gradients lies in the exponential curvature of this gradient. L is sensitive to small changes in u, which makes it difficult to optimize it with respect to Θ_e on high learning rates.

The instability of the gradient w.r.t. to u demands an adaptive learning rate for encoder parameters Θ_u that can adapt to sudden large changes in $\nabla_u L$.

We now propose that this adaptive learning rate can be achieved by applying BachNorm (BN) (Ioffe and Szegedy, 2015) transformation to f_u. BN transformation for an incoming mini-batch $\{u_{i=1}^m\}$ is given by

$$u_{BN} = \gamma \frac{u - \mu_{\text{batch}}}{\sqrt{\sigma^2_{\text{batch}+\epsilon}}} + b. \quad (5)$$

Here, $\mu_{\text{batch}} = \frac{1}{m}\sum_{i=1}^m u_i$, $\sigma^2_{\text{batch}} = \frac{1}{m}\sum_{i=1}^m (u_i - \mu_{\text{batch}})^2$, γ is the gain parameter and finally b is the shift parameter. We are specifically interested in the scaling factor $\frac{\gamma}{\sqrt{\sigma^2_{\text{batch}}}}$, because the sample variance grows and shrinks with large changes in the norm of the mini-batch therefore allowing the scaling factor to approximately dictates the norm of the activations. Let L be defined as before, the posterior q is now a function of u_{BN}. The gradients w.r.t. u and the gain parameter γ are

$$\nabla_u L = \frac{\gamma}{\sqrt{\sigma^2_{\text{batch}+\epsilon}}} P_u \nabla_{u_{BN}} L \quad (6)$$

$$\nabla_\gamma L = \frac{(u - \mu_{\text{batch}})}{\sqrt{\sigma^2_{\text{batch}+\epsilon}}} . \nabla_{u_{BN}} L, \quad (7)$$

where P_u is a projection matrix. If $\nabla_{u_{BN}} L$ is large with respect to the out-going u_{BN}, the scaling term brings it down. Therefore, the scaling term works like an adaptive learning rate that grows and shrinks in response to the change in norm of the batch of u's due to large gradient updates to the weights, thus resolving the issue with the diverging gradients. As shown in figure 3, after applying BN to u (red), the KL term minimizes fairly slowly compared to the case (blue) when no BN is applied to f_u. This was also observed by (Srivastava and Sutton, 2017). We experimentally found that at this point the topics start to improve when the learning rate is ≥ 0.001.

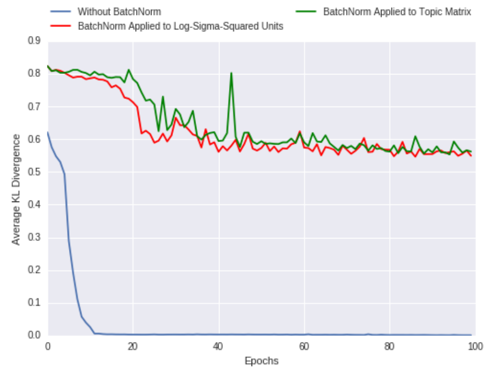

Figure 3: In optimization without any BatchNorm, the average KL gets minimized fairly early in the training. With BatchNorm applied to the encoder unit that produces $\log \sigma^2$, the KL minimization is slow and slower if BatchNorm is also applied to each of the topics in the decoder.

In order to establish that the improvement in training comes from the adaptive learning rate property of the gain parameter we replace the divisor in the BN transformation with the ℓ_2 norm of the activation. We neither center the activations nor apply any shift to them. This normalization performs equivalently and occasionally better than BN, therefore confirming our hypothesis. It also removes any dependency on batch-level statistics that might be a requirement in models that make i.i.d assumptions.

Component Collapsing

Another well known issue in VAEs such as dnPAM is the problem of component collapsing (Dinh and Dumoulin, 2016; van den Oord et al., 2017). In the context of topic models, component collapsing is a bad local minimum of VAEs in which the model only learns a small number of topics out of K (Srivastava and Sutton, 2017). For example, suppose we train a 3-PAM model on the Omniglot dataset (Lake et al., 2015) using the stochastic variational inference from Kingma and Welling (2013). Figure 2 shows nine randomly sampled topics for from this model which have been reshaped to Omniglot image dimensions. All the topics look exactly the same, with a few exceptions. This is clearly not a useful set of topics.

When trained without applying BN to the u, the KL terms across most of the latent dimensions (components of z) vanish to zero. We call them collapsed dimensions, since the posterior along them has collapsed to the prior. As a result, the decoder only receives the sampling noise along such collapsed dimensions and in order to minimize the noise in the output, it makes the weights corresponding to these collapsed components very small. In practice this means that these weight do not participate in learning and therefore do not represent any meaningful topic.

Following Srivastava and Sutton (2017), we also found that the topic coherence increases drastically when BN is also applied to the topic matrix prior to the application of the softmax nonlinearity along with f_u. Besides preventing the softmax units to saturate, this slows down the KL minimization further as shown by the green curve in figure 3.

5 Experiments and Results

We evaluate how dnPAM inference performs for different architectures of PAM models when compared to the state-of-art collapsed Gibbs inference. To this end we evaluate three different PAM architectures, 4-PAM, 5-PAM and MoLDA, on two different datasets, 20 Newsgroups and NIPS abstracts (Lichman, 2013). We use these two data sets because they represent two extreme settings. 20 Newsgroup is a large dataset (12,000 documents) but with a more restricted vocabulary (2000 words) whereas the NIPS dataset is smaller in size (1500 abstracts) dataset but has a considerably larger vocabulary (12419 words). We compare inference methods both on time required for training as well as topic quality. As a measure of topic quality, we use the topic coherence metric (normalized point-wise mutual information), which as shown in Lau et al. (2014) corresponds very well with human judgment on the quality of topics. We do not report perplexity of the models because it has been repeatedly shown to not be a good measure of topic coherence and even to be negatively correlated with the topic quality in some cases (Lau et al., 2014; Chang et al., 2009; Srivastava and Sutton, 2017).

We start by comparing the topic coherence across the different topic models on the 20 Newsgroup dataset. We train an LDA model using both collapsed Gibbs sampling[3] (Griffiths and Steyvers, 2004) and Decoupled Mean-Field Variational Inference (DMFVI)[4] (Blei et al., 2003). Using Mallet, we train a 4-PAM model using 10000 iterations of collapsed Gibbs sampling and using dnPAM we train a 4-PAM, a 5-PAM, a MoLDA and a correlated topic model (CTM). In this experiment we use 50 sub-topics for all models[5]. For 4-PAM and 5-PAM, we use two super-topics following (Li and McCallum, 2006), and two additional super-duper-topics for 5-PAM. As shown in Table 1 all PAM models perform better than LDA-type models, showing that more complex PAM architectures do improve the quality of the topics. Additionally 4-PAM and 5-PAM models trained on dnPAM beat all the LDA models for topic quality. MoLDA and CTM trained using dnPAM also perform competitively with the LDA models but the CTM model falls significantly behind PAM models on topic coherence.

Next, to study the effect of increasing the number of super and sub-topics in PAM models on the topic quality we increase the number of super-duper-topics to 10, super-topics to 50 and sub-topics to 100 and re-run the previous experiment but only on the 4-PAM or deeper models. Table 2 shows the topic coherence for each of these models and also the training time. Not only our inference method produces better topics it also is an order of magnitude faster than the state-of-art Gibbs sampling based inference for 4-PAM. Note that we run the sampler for a total of 3000 iterations with the burn-in parameter set to 2000 iterations.

For the smaller NIPS dataset, we repeat the same experiments only for the PAM models again under the same exact settings as described above. Reported in table 3 are the topic coherence for smaller PAM models with 50 sub-topics. Again, we allowed 10,000 Gibbs iterations which took more than a day to finish but did not beat dnPAM-trained models on topic quality. For the bigger PAM models we replicated the experiments from the original paper. As reported in table 4 we found that while the collapsed Gibbs based 4-PAM model produced the best topics it did so in 15 hours. On the other hand, dnPAM-trained models produced topics with comparable quality and only took a fraction of the inference time for Gibbs by being able to leverage the GPU architecture for computing dot-products very efficiently. We are not aware of GPU-based implementations of other inference method for PAMs.

[3] We used the Mallet implementation (McCallum, 2002).

[4] We used the scikit-learn implementation (Pedregosa et al., 2011).

[5] For MoLDA we use 10 and 50 topics

Table 1: Topic Coherence on 20Newsgroup for 50 topics. PAM models use two super-topics at each level. For MoLDA, we report separate the coherence for each component in the admixture.

				dnPAM				
	LDA GIBBS	**LDA DMFVI**	**4-PAM GIBBS**	**4-PAM**	**5-PAM**	**CTM**	**MoLDA**	
							10	**50**
Topic Coherence	0.17	0.11	0.20	**0.24**	**0.24**	0.14	0.29	0.21

Table 2: Topic coherence for models trained on 20Newsgroup dataset for for 100 topics with 50 super-topics.

		dnPAM			
	4-PAM GIBBS	**4-PAM**	**5-PAM**	**MoLDA**	
				50	**100**
Topic Coherence	0.19	**0.22**	0.21	0.24	0.21
Training Time (Min.)	594	11	16	16	

Table 3: Topic coherence for models trained on NIPS dataset for 50 topics with 2 super-topics.

		dnPAM			
	4-PAM GIBBS	**4-PAM**	**5-PAM**	**MoLDA**	
				10	**50**
Topic Coherence	0.033	**0.042**	0.039	0.036	0.024

Table 4: Topic coherence for models trained on NIPS dataset for 100 topics with 50 super-topics.

		dnPAM			
	4-PAM GIBBS	**4-PAM**	**5-PAM**	**MoLDA**	
				50	**100**
Topic Coherence	**0.047**	0.041	0.045	0.025	0.024
Training Time (Min.)	892	19	26	11	

5.1 Hyper-Parameter Tuning

For the experiments in this section we did not conduct an extensive hyper-parameter tuning. We did a grid search for setting the encoder capacity while moving between the two datasets. As a general guideline for PAM models, the encoder capacity should grow with the vocabulary size. For the learning rate, we used the default setting of $1e^{-3}$ for the Adam optimizer for all the models. We used a batch-size of 200 for 20 Newsgroup dataset as used in (Srivastava and Sutton, 2017) and 50 for the NIPS dataset. Though we found that the topic coherence, especially for the smaller NIPS dataset is sensitive to the batch-size setting and initialization. For certain settings we were able to achieve higher topic coherence than the average topic coherence reported in this section.

6 Related Work

Topic models have been explored extensively via directed (Blei et al., 2003; Li and McCallum, 2006; Blei and Lafferty, 2006; Blei et al., 2004) as well as undirected models or restricted Boltzmann machines (Larochelle and Lauly, 2012; Hinton and Salakhutdinov, 2009). Hierarchical extensions to these models have received special attention since they allow capturing the correlations between the topics and provide meaningful interpretation to the latent structures in the data.

Recent advancements in blackbox-type inference method (Kucukelbir et al., 2016; Ranganath et al., 2014; Mnih and Gregor, 2014) have made it easier to try newer models without the need of deriving model-specific inference algorithms.

7 Conclusion

In this work we introduced dnPAM, which generalizes the idea of amortized variational inference that VAEs provide to a large class of deep topic models.

References

David Blei and John Lafferty. 2006. Correlated topic models. *Advances in neural information processing systems* 18:147.

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

D.M. Blei, T.L. Griffiths, M.I. Jordan, and J.B. Tenenbaum. 2004. Hierarchical Topic Models and the Nested Chinese Restaurant Process. *Advances in Neural Information Processing Systems 16: Proceedings of the 2003 Conference* .

Jonathan Chang, Jordan L Boyd-Graber, Sean Gerrish, Chong Wang, and David M Blei. 2009. Reading tea leaves: How humans interpret topic models. In *Nips*. volume 31, pages 1–9.

Laurent Dinh and Vincent Dumoulin. 2016. Training neural Bayesian nets.

Thomas L Griffiths and Mark Steyvers. 2004. Finding scientific topics. *Proceedings of the National academy of Sciences* 101(suppl 1):5228–5235.

Geoffrey E Hinton and Ruslan R Salakhutdinov. 2009. Replicated softmax: an undirected topic model. In *Advances in neural information processing systems*. pages 1607–1614.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of The 32nd International Conference on Machine Learning*. pages 448–456.

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* .

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* .

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. 2016. Automatic differentiation variational inference. *arXiv preprint arXiv:1603.00788* .

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. 2015. Human-level concept learning through probabilistic program induction. *Science* 350(6266):1332–1338.

Hugo Larochelle and Stanislas Lauly. 2012. A neural autoregressive topic model. In *Advances in Neural Information Processing Systems*. pages 2708–2716.

Jey Han Lau, David Newman, and Timothy Baldwin. 2014. Machine reading tea leaves: Automatically evaluating topic coherence and topic model quality. In *EACL*. pages 530–539.

Wei Li, David Blei, and Andrew McCallum. 2012. Nonparametric bayes pachinko allocation. *arXiv preprint arXiv:1206.5270* .

Wei Li and Andrew McCallum. 2006. Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of the 23rd international conference on Machine learning*. ACM, pages 577–584.

M. Lichman. 2013. UCI machine learning repository. `http://archive.ics.uci.edu/ml`.

Andrew Kachites McCallum. 2002. Mallet: A machine learning for language toolkit. `http://mallet.cs.umass.edu`.

David Mimno, Wei Li, and Andrew McCallum. 2007. Mixtures of hierarchical topics with pachinko allocation. In *Proceedings of the 24th international conference on Machine learning*. ACM, pages 633–640.

Andriy Mnih and Karol Gregor. 2014. Neural variational inference and learning in belief networks. *arXiv preprint arXiv:1402.0030* .

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Rajesh Ranganath, Sean Gerrish, and David M Blei. 2014. Black box variational inference. In *AISTATS*. pages 814–822.

Danilo Rezende and Shakir Mohamed. 2015. Variational inference with normalizing flows. In *Proceedings of The 32nd International Conference on Machine Learning*. pages 1530–1538.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of The 31st International Conference on Machine Learning*. pages 1278–1286.

Francisco R Ruiz, Michalis Titsias RC AUEB, and David Blei. 2016. The generalized reparameterization gradient. In *Advances in Neural Information Processing Systems*. pages 460–468.

Akash Srivastava and Charles Sutton. 2017. Autoencoding variational inference for topic models. *International Conference on Learning Representations (ICLR)* .

Aaron van den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*. pages 6297–6306.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8(3-4):229–256.