LEARNING LATENT SUPERSTRUCTURES IN VARIATIONAL AUTOENCODERS FOR DEEP MULTIDIMENSIONAL CLUSTERING

Anonymous authors
Paper under double-blind review

ABSTRACT

We investigate a variant of variational autoencoders where there is a superstructure of discrete latent variables on top of the latent features. In general, our superstructure is a tree structure of multiple super latent variables and it is automatically learned from data. When there is only one latent variable in the superstructure, our model reduces to one that assumes the latent features to be generated from a Gaussian mixture model. We call our model the latent tree variational autoencoder (LTV AE). Whereas previous deep learning methods for clustering produce only one partition of data, LTV AE produces multiple partitions of data, each being given by one super latent variable. This is desirable because high dimensional data usually have many different natural facets and can be meaningfully partitioned in multiple ways.

1 INTRODUCTION

Clustering is a fundamental task in unsupervised machine learning, and it is central to many data-driven application domains. Cluster analysis partitions all the data into disjoint groups, and one can understand the structure of the data by examining examples in each group. Many clustering methods have been proposed in the literature, such as k-means [MacQueen et al. (1967)], Gaussian mixture models [Christopher (2016)] and spectral clustering [Von Luxburg (2007)]. Conventional clustering methods are generally applied directly on the original data space. However, it is challenging to perform cluster analysis on high dimensional and unstructured data [Steinbach et al. (2004)], such as images. It is not only because the dimensionality is high, but also because the original data space is too complex to interpret, e.g. there are semantic gaps between pixel values and objects in images.

Recently, deep learning based clustering methods have been proposed that simultaneously learn non-linear embeddings through deep neural networks and perform cluster analysis on the embedding space. The representation learning process learns effective high-level representations from high dimensional data and helps the cluster analysis. This is typically achieved by unsupervised deep learning methods, such as restricted Boltzmann machine (RBM) [Hinton et al. (2006); Hinton & Salakhutdinov (2006)], autoencoders (AE) [Vincent et al. (2008; 2010)], variational autoencoders (VAE) [Kingma & Welling (2014)], etc. A simple two-stage approach is to first learn low-dimensional embeddings using these methods, and then perform cluster analysis on the embeddings. However, without any supervision, the representation learning do not necessarily reveal the true cluster structure of the data. DEC [Xie et al. (2016)] is a method that simultaneously learns feature representations and cluster assignments through deep autoencoders. It gradually improves the clustering by driving the deep network to learn a better mapping. Variational deep embedding [Jiang et al. (2017)] is a generative method that models the data generative process using a Gaussian mixture model combined with a VAE, and also performs joint learning of representations and clustering. Similarly, GMVAE [Dilokthanakul et al. (2016)] performs joint learning of a GMM and a VAE, but instead generates the mixture components through neural networks. Deep clustering network network (DCN) [Yang et al. (2017)] is another one that jointly learns an autoencoder and performs k-means clustering. These joint learning methods consistently achieve better clustering results than conventional ones.
Previous deep learning based clustering methods assume one single partition over the data and that all attributes define that partition. In real-world applications, however, the assumptions are usually not true. High-dimensional data are often multifaceted and can be meaningfully partitioned in multiple ways based on subsets of attributes [Chen et al. (2012)]. For example, a student population can be clustered in one way based on course grades and in another way based on extracurricular activities. Movie reviews can be clustered based on both sentiment (positive or negative) and genre (comedy, action, war, etc.). It is challenging to discover the multi-facet structures of data, especially for high-dimensional data.

To resolve the above issues, we propose an unsupervised learning method, \textit{latent tree variational autoencoder} (LTV AE) to learn latent superstructures in variational autoencoders, and simultaneously perform representation learning and structure learning. LTVAE is a generative model, where the data is assumed to be generated from latent features through neural networks, while the latent features themselves are generated from tree-structured Bayesian networks with another level of latent variables as shown in Fig. [1] Each of those latent variables defines a facet of clustering. The proposed method automatically selects subsets of latent features for each facet, and learns the dependency structure among different facets. This is achieved through systematic structure learning. Consequently, LTV AE is able to discover complex structures of data rather than one partition. We also propose efficient learning algorithms for LTV AE with gradient descent and Stepwise EM through message passing.

The rest of the paper is organized as follows. The related works are reviewed in Section 2. We introduce the proposed method and learning algorithms in Section 3. In Section 4, we present the empirical results. The conclusion is given in Section 5.

2 RELATED WORKS

Clustering has been extensively studied in the literature in many aspects [Aggarwal & Reddy (2013)]. More complex clustering methods related to structure learning using Bayesian nonparametrics have been proposed, like Dirichlet Process [Blei et al. (2006), Hierarchical Dirichlet Process (HDP) Teh et al. (2006). However, those are with conventional clustering methods that apply on raw data. Recently, deep learning based clustering methods have drawn more and more attention. Besides the methods mentioned in the previous section, there are some other works related to the topic of interest. Improved Deep Embedded Clustering [Guo et al. (2017)] improves DEC by keeping the decoder network and adding reconstruction loss to the original clustering loss in DEC. The method proposed in [Yang et al. (2016)] uses convolutional neural networks and jointly learns the representations and clustering in a recurrent framework. All these methods assume flat partitions over the data, and do not attempt the structure learning issue. An exception is hierarchical nonparametric variational autoencoders proposed in [Goyal et al. (2017)]. It uses nCRP as the prior for VAE to allow infinitely deep and branching tree hierarchy structure and focuses on learning hierarchy of concepts. Different from it, our work focuses on multifacets of clustering.

3 THE PROPOSED METHOD

In this section, we present the proposed \textit{latent tree variational autoencoder} and the learning algorithms for joint representation learning and structure learning for multidimensional clustering.

3.1 LATENT TREE VARIATIONAL AUTOENCODER

Deep generative models assume that data \( x \) is generated from latent continuous variable \( z \) through some random process. The process consists of two steps: (1) a value \( z \) is generated from some prior distribution \( p(z) \); (2) the observation \( x \) is generated from the conditional distribution \( p_\theta(x|z) \), which is parameterized through deep neural networks. Thus, it defines the joint distribution between observation \( x \) and latent variable \( z \):

\[
p(x, z) = p(z)p_\theta(x|z)
\]

This process is hidden from our view, and we learn this process by maximizing the marginal log-likelihood \( p(x) \) over the parameters \( \theta \) and latent variable \( z \) from data. After the learning, the latent
variable $z$ can be regarded as the deep representations of $x$ since it captures the most relevant information of $x$. Thus, the learning process is also called representation learning.

In order to learn the latent structure of $z$, for example multidimensional cluster structure, we introduce a set of latent variables $Y_1, ..., Y_l$ on top of $z$. A single $z$ or multiple $z$’s form a node $z_b$. Suppose variables in $z$ form $B$ nodes of $z_1, \ldots, z_B$. Each latent variable $Y$ is only connected to a subset of nodes, and the dependency of each $z_b$ and its parent $Y$ is characterized by a conditional Gaussian distribution. Furthermore, the latent variables $Y_1, ..., Y_l$ are connected to each other, and the dependency of a latent variable $Y$ on its parent $Y'$ is characterized by a conditional distribution $P(Y|Y')$. This essentially forms a Bayesian network. And if we restrict the network to be tree-structured, the $z$ and $Y$ together form a latent tree model [Zhang 2004, Poon et al. 2010, 2013; Mourad et al. 2013; Pearl 2014; Zhang & Poon 2017] with $z$ being the observed variables and $Y$ being the latent variables. For multidimensional clustering, each latent variable $Y$ is taken to be a discrete variable, where each discrete state $y$ of $Y$ defines a cluster. Each latent variable $Y$ thus defines a facet partition over the data based on subset of attributes and multiple $Y$’s define multiple facets. Given a value $y$ of $Y$, $z_b$ follows a conditional Gaussian distribution $P(z_b|y) = N(\mu_y, \Sigma_y)$ with mean vector $\mu_y$ and covariance matrix $\Sigma_y$. Thus, each $z_b$ and its parent constitute a Gaussian mixture model (GMM). Suppose the parent of a node is denoted as $\pi(\cdot)$, the marginal distribution of $z$ is defined as follows

$$p(z) = \sum_{Y} \prod_{j=1}^{l} p(y_j|\pi(Y_j)) \prod_{b=1}^{B} N(z_b|\mu_{\pi(z_b)}, \Sigma_{\pi(z_b)}),$$

which sums over all possible combinations of $Y$ states. As a matter of fact, a GMM is a Gaussian LTM that has only one latent variable connecting to all observed variables.

Let the latent structure of $Y$ be $S$, defining the number of latent variables in $Y$, the number of discrete states in each variable $Y$ and the connectivity structure among all variables in $z$ and $Y$. And let the parameters for all conditional probabilities in the latent structure be $\Theta$. Both the latent structure $S$ and the latent parameters $\Theta$ are unknown. We aim to jointly learn data representations and the latent structure. The proposed LTVAE model is shown in Fig. 1. The latent structure $S$ is automatically learned from data and will be discussed in a later section.

Due to the existence of generation network, the inference of the model is intractable. Instead, we do variational inference for the latent variable $z$ by introducing an inference network [Kingma & Welling 2014] and define an approximate posterior $q_\phi(z|x)$. Thus, the marginal loglikelihood of the data given $(S, \Theta)$ is:

$$\mathcal{L}_{\text{ELBO}}(x) = \mathbb{E}[q_\phi(z|x)] + \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] + \mathbb{E}_{q_\phi(z|x)}[\log \sum_y p_S(z, y; \Theta)],$$

where $\mathcal{L}_{\text{ELBO}}(x)$ is defined as the evidence lower bound (ELBO). $\log \sum_y p_S(z, y; \Theta)$ is the marginal loglikelihood of the latent variable $z$ under the latent tree model. Note how complicated it can be due to possible structures of $S$. $p_\theta(x|z)$ could be a Gaussian distribution if the input data is real-valued, or a Bernoulli distribution if binary, parameterized by the generation network. Using SGVB.
sampling, the ELBO can be asymptotically estimated by
\[
\mathcal{L}_{\text{ELBO}}(x) \approx \mathbb{E}[q_{\phi}(z|x)] + \frac{1}{M} \sum_{i=1}^{M} \log p_{\theta}(x|z^{(i)}) + \log \sum_{y} p_{S}(z^{(i)}, y; \Theta)
\] (4)
where \( z^{(i)} \sim q_{\phi}(z|x) \). The term \( \mathbb{E}[q_{\phi}(z|x)] \) can be computed analytically if we choose the form of \( q_{\phi}(z|x) \) to be a Gaussian distribution \( \mathcal{N}(z; \mu_x, \sigma_x) \): \( \mathbb{E}[q_{\phi}(z|x)] = J \log(2\pi) + \frac{1}{2} \sum_{j=1}^{J}(1 + \log \sigma_j^2) \), where \( J \) is the dimensionality of \( z \).

Furthermore, the marginal loglikelihood \( \log \sum_{y} p_{S}(z^{(i)}, y; \Theta) \) can be computed efficiently through message passing. Message passing is an efficient algorithm for inference in Bayesian networks [Koller & Friedman, 2009; Poon et al., 2013]. In message passing, we first build a clique tree using the factors in the defined probability density. Because of the tree structure, each \( z_b \) along with its parent form a clique with the potential \( \psi(z_b, y) \) being the corresponding conditional distribution. This is illustrated in Fig. 2. With the sampled \( z^{(i)} \), we can compute the message \( \psi'(y) \) by absorbing the evidence from \( z \). During collecting message phase, the message \( \psi'(y) \) are sent towards the pivot. After receiving all messages, the pivot distributes back messages towards all \( z \). Both the posterior of \( Y \) and the marginal loglikelihood of \( z^{(i)} \) thus can be computed in the final normalization step.

3.2 Parameter Learning through Gradient Descent and Stepwise EM with Message Passing

In this section, we propose efficient learning algorithms for LTVAE through gradient descent and stepwise EM with message passing.

Given the latent tree model \((S, \Theta)\), the parameters of neural networks can be efficiently optimized through stochastic gradient descent (SGD). However, in order to learn the model, it is important to efficiently compute the gradient of the marginal loglikelihood \( \log p_{S}(z; \Theta) \) from the latent tree model, the third term in Eq. 4. Here, we propose an efficient method to compute gradient through message passing. Let \( z_b \) be the variables that we want to compute gradient with respect to, and let \( Y_b \) be the parent node. The marginal loglikelihood of full \( z \) can be written as
\[
p_{S}(z; \Theta) = \sum_{y_b} \mathcal{N}(z_b; \mu_{y_b}, \Sigma_{y_b}) f(y_b),
\] (5)
where \( \pi(\cdot) \) denote the parent node of current node and \( f(y_b) \) is the collection of all the rest of the terms not containing \( z_b \). The gradient w.r.t \( z_b \) thus can be computed as
\[
g_{z_b} = \left. \frac{1}{p(z)} \frac{\partial \sum_{y_b} f(y_b) \mathcal{N}(z_b; \mu_{y_b}, \Sigma_{y_b})}{\partial z_b} \right|_{z_b} = \sum_{y_b} p(y_b|z) \frac{\partial \log f(y_b) \mathcal{N}(z_b; \mu_{y_b}, \Sigma_{y_b})}{\partial z_b} = \sum_{y_b} p(y_b|z) \Sigma_{y_b}^{-1} (\mu_{y_b} - z_b)
\] (6)
where \( p(y_b|z) \) is the posterior probability of \( y_b \) and can be computed efficiently with message passing as described in the previous section. Since \( z = [z_1, \ldots, z_B] \), we have
\[
\frac{\partial \log p(z)}{\partial z} = \left[ \frac{\partial \log p(z)}{\partial z_1}, \ldots, \frac{\partial \log p(z)}{\partial z_B} \right] = [g_{z_1}, \ldots, g_{z_B}].
\] (7)

With the efficient computation of the third term in Eq. 4 and its gradient w.r.t \( z \) through message passing, the parameters of inference network and generation network can be efficiently optimized through SGD.

In order to jointly learn the parameters of the latent tree \( \Theta \), we propose Stepwise EM algorithm based on mini-batch of data. Specifically, we maximize the third term in Eq. 4 i.e. the marginal loglikelihood of \( z \) under the latent tree. In the Stepwise E-step, we compute the distributions \( P(y, y'|z, \theta^{(t-1)}) \) and \( P(y|z, \theta^{(t-1)}) \) for each latent node \( Y \) and its parent \( Y' \). In the Stepwise M-step, we estimate the new parameter \( \theta^{(t)} \). Let \( s(z, y) \) be a vector the sufficient statistics for a single data case. Let \( \bar{s} = \mathbb{E}_{p_{\theta}(y|z, \bar{\Theta})} [s(z, y)] \) be the expected sufficient statistics for the data case, where the expectation is w.r.t the posterior distribution of \( y \) with current parameter. And let
The cardinalities of latent variables, the connectivities among variables. We aim at finding the model
structure with a high BIC score. We use the hill-climbing algorithm to search for
a structure with a high BIC score. We use the hill-climbing algorithm to search for
the model fit and model complexity. To this end, we perform systematic searching to find
the model fit and model complexity. To this end, we perform systematic searching to find
the best one. Acceleration techniques Poon et al. (2013) are adopted that make the algorithm efficient
enough. The structure learning is performed iteratively together with the parameter learning of
neural networks.

3.3 Structure Learning

For the latent structure $S$, there are four aspects need to determine: the number of latent variables,
the cardinalities of latent variables, the connectivities among variables. We aim at finding the model
that maximizes the BIC score [Schwarz et al. (1978); Koller & Friedman (2009)]:

$$BIC(m|D) = \log P(D|m, \theta^*) - \frac{d(m)}{2} \log N,$$

where $\theta^*$ is the MLE of the parameters and $d(m)$ is the number of independent parameters. The
first term is known as the likelihood term. It favors models that fit data well. The second term is
known as the penalty term. It discourages complex models. Hence, the BIC score provides a trade-off
between model fit and model complexity. To this end, we perform systematic searching to find
a structure with a high BIC score. We use the hill-climbing algorithm to search for $m^*$ as in Poon
et al. (2010; 2013), and define 5 search operators: node introduction (NI) and node deletion (ND)
to introduce new latent nodes and delete existing nodes, state introduction (SI) and state deletion
(SD) to add a new state and delete a state for existing nodes, node relocation (NR) to change links
of existing nodes, pouching (PO) and unpouching (UP) operators to combine nodes into a single
node and separate variables from a node. The structure search operators are shown in Fig. 3. Each
operator produces a set of candidates from existing structure, and the best candidate is picked if it
improves the previous one. To reduce the number of possible search candidates, we first perform
SI, NI and PO to expand the structure and pick the best model. Then we perform NR to adjust the
best model. Finally, we perform UP, ND and SD to simplify the current best structure and pick the
best one. Acceleration techniques Poon et al. (2013) are adopted that make the algorithm efficient
enough. The structure learning is performed iteratively together with the parameter learning of
neural networks.

The overall learning algorithm is illustrated in Algorithm 1. Starting from a pretrained model, we it-
eratively improve the structure and parameters of latent tree model while learning the representations
of data through neural network in a greedy manner. Using current structure $S^t$ as the initial structure,
we search for a better model. With new latent tree model, we optimize for a better representation
until convergence.
the data points have two facets

$Y$

We first demonstrate the effectiveness of the proposed method through synthetic data. Assume that

Algorithm 1

Learning Latent Tree Variational Autoencoder

Input: data $D$, $z$ dim, neural networks, $E$

$\theta, \phi, S^0, \Theta^0 \leftarrow \text{pretrain}(D)$

repeat

for $e = 1$ to $E$ do

for each minibatch $\mathcal{X}$ in $D$ do

Compute $q_\phi(z|x, \sigma)$

Sample $z(i) \sim q(z|x, \sigma)$

Compute log $p_D(z; \Theta)$ and $\log p_E(z)$ from Eq. 5 and 7

Compute ELBO from Eq. 4

$\theta, \phi \leftarrow \text{Back-propagation and SGD step}$

$\Theta \leftarrow \text{StepwiseEM}(z(i))$

end for

end for

$D_{z} \leftarrow \mu_D$

repeat

$S^t, \Theta^t \leftarrow \text{SearchWith}(S^{t-1}, \Theta^{t-1}, \{SI, NI, PO\})$

$S^t, \Theta^t \leftarrow \text{SearchWith}(S^t, \Theta^t, \{NR\})$

$S^t, \Theta^t \leftarrow \text{SearchWith}(S^t, \Theta^t, \{UP, ND, SD\})$

until $BIC(S^t, \Theta^t|D_{z}) \leq BIC(S^{t-1}, \Theta^{t-1}|D_{z})$

until stopping criteria

return $\theta, \phi, S, \Theta$

4 Experiments

4.1 Synthetic-Data Demonstration

We first demonstrate the effectiveness of the proposed method through synthetic data. Assume that the data points have two facets $Y_1$ and $Y_2$, where each facet controls a subset of attributes (e.g. two-dimensional domain) and defines one partition over the data. This four-dimensional domain $Z = \{z_1, z_2, z_3, z_4\}$ is a latent representation which we do not observe. What we observe is $X \in \mathbb{R}^{100}$ that is obtained via the following non-linear transformation:

$x = \sigma(U\sigma(Wz)),$

where $W \in \mathbb{R}^{10 \times 2}$ and $U \in \mathbb{R}^{100 \times 10}$ are matrices whose entries follow the zero-mean unit-variance i.i.d. Gaussian distribution. $\sigma(\cdot)$ is a sigmoid function to introduce nonlinearity. The generative model is shown in Fig. 4(a). We define two clusters in facet $Y_1$ and two clusters in facet $Y_2$, and generate 5,000 samples of $x$. Under the above generative model, recovering the two facets $Y_1$ and $Y_2$ structure and the latent $z$ domain from the observation of $x$ seems very challenging. All previous DNN-based methods (AE+GMM, DEC, DCN, etc.) are only able to discover one-facet of clustering (i.e. one partition over the data), and none of these is applicable to solve such a multidimensional clustering problem. Fig. 4(b) shows the results of the proposed method. As one can see, the LTVAE successfully discovers the true superstructure of $Y_1$ and $Y_2$. The 2-d plot of $z_1$ and $z_2$ shows the separable latent space clusters under facet $Y_1$, and it matches the ground-truth cluster assignments. Additionally, the 2-d plot of $z_3$ and $z_4$ shows another separable clusters under facet $Y_2$, and it also matches the ground-truth cluster assignments well in the other facet.

4.2 Real-Data Experiment Setup

We evaluate the proposed LTVAE model on two image datasets and two other datasets, and compare it against other deep learning based clustering algorithms, including two-stage methods, AE+GMM and VAE+GMM, which first learn AE/VAE [Kingma & Welling, 2014] models then construct a GMM on top of them, and joint learning methods, DEC [Xie et al., 2016] and DCN [Yang et al., 2017]. The datasets include MNIST, STL-10, Reuters [Xie et al., 2016; Jiang et al., 2017] and the Heterogeneity Human Activity Recognition (HHAR) dataset. When evaluating the clustering performance, for fair of comparison, we follow previous works [Xie et al., 2016; Yang et al., 2017] and use the network structures of $d - 500 - 500 - 2000 - 10$ for the encoder network and $10 - 2000 - 500 - 500 - d$ for the decoder network for all datasets, where $d$ is the data-space dimension, which varies among datasets. All layers are fully-connected. We follow the pretraining procedure as in [Xie
The most important features of the proposed model are that it can perform variable selection for model-based clustering, leading to multiple facets clustering.

We use the standard unsupervised evaluation metric and protocols for evaluations and comparisons to other algorithms [Yang et al. (2010)]. For baseline algorithms we set the number of clusters to the number of ground-truth categories. While for LTVAE, it automatically determines the number of facets and latent superstructure through structure learning. We evaluate performance with
Figure 5: Two facet clustering results from LTVAE are in (a) digit identity and (b) shape and pose. Each row contains the top 10 scoring elements from one cluster. (c) shows the pose variations by fixing the digit cluster in facet 1 and changing the cluster in facet 2. It can be seen that up-right, left-tilted and right-tilted images of the same digits are clearly recognizable.

Table 2: Clustering Accuracy of clustering results.

<table>
<thead>
<tr>
<th>Model</th>
<th>MNIST</th>
<th>STL-10</th>
<th>Reuters</th>
<th>HHAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>AE+GMM</td>
<td>82.18%</td>
<td>79.83%</td>
<td>68.68%</td>
<td>78.90%</td>
</tr>
<tr>
<td>VAE+GMM</td>
<td>76.87%</td>
<td>79.49%</td>
<td>65.85%</td>
<td>67.91%</td>
</tr>
<tr>
<td>DEC</td>
<td>84.30%</td>
<td>80.62%</td>
<td>74.32%</td>
<td>79.86%</td>
</tr>
<tr>
<td>DCN</td>
<td>83.32%</td>
<td>85.88%</td>
<td>75.05%</td>
<td>81.26%</td>
</tr>
<tr>
<td>LTVAE</td>
<td>86.32%</td>
<td>90.00%</td>
<td>80.96%</td>
<td>85.00%</td>
</tr>
</tbody>
</table>

unsupervised clustering accuracy (ACC):

\[
ACC = \max_m \sum_{i=1}^{n} \frac{1}{n} \{l_i = m(c_i)\},
\]

where \( l_i \) is the groundtruth label, \( c_i \) is the cluster assignment produced by the algorithm, and \( m \) ranges over all possible mappings between clusters and labels. Table 2 show the quantitative clustering results compared with previous works. With \( z \) dimension of small value like 10, LTVAE usually discovers only one facet. It can be seen the, for MNIST dataset LTVAE achieves clustering accuracy of 86.32%, better than the results of other methods. This is also the case for STL-10, Reuters and HHAR.

More importantly, the proposed LTVAE does not just give one partition over the data. Instead, it explains the data in multi-faceted ways. Unlike previous clustering experiments, for this experiment, we choose the \( z \) dimension to be 20. Fig. 5 shows the two facet clustering results for MNIST. It can be seen that facet 1 gives quite clean clustering over the identity of the digits and the ten digits are well separated. On the other hand, facet 2 gives a more grand partition based on the shape and pose. Note how up-right “4” and “9” are similar, and how tilted “4”, “7” and “9” are similar. The facet meanings are more evident in Fig. 5(c). Fig. 6 shows four facets discovered for the STL-10 dataset. Although it is hard to characterize precisely how the facets differ from each other, there are visible patterns. For example, the cats, monkeys and birds in facet 2 have clearly visible eyes, while this is not always true in facet 1. The deers in facet 2 are all showing their antlers/ears, while this is not true in facet 3. In facet 2 we see frontal views of cars, while in facets 1 and 3 we see side view of cars. In facet 1, each cluster consists of the same types of objects/animal. In facet 3/4, images in the same cluster do not necessarily show the same type of objects/animals. However, they have similar overall feel.
4.5 IMAGE GENERATION

Since the structure of the data in latent space is automatically learned through the latent tree, we can sample the data in a more structured way. One way is through ancestral sampling, where we first sample the root of the latent tree and then hierarchically sample the children variables to get $z$, from which the images can be generated through generation network. The other way is to pick one component from the MoG and sample $z$ from that component. This produces samples from a particular cluster. Fig. 7 shows the samples generated in this way. As it can be seen, digits sampled from each component has clear semantic meaning and belong to the same category. Whereas, the samples generated by VAE does not have such structure.

5 CONCLUSIONS

In this paper, we propose an unsupervised learning method, latent tree variational autoencoder (LTVAE), which simultaneously performs representation learning and multidimensional clustering. Different from previous deep learning based clustering methods, LTVAE learns latent embeddings from data and discovers multi-facet clustering structure based on subsets of latent features rather than one partition over data. Experiments show that the proposed method achieves state-of-the-art clustering performance and realz reasonable multifacet structures of the data.

REFERENCES


