Deep Infinite Mixture Latent Variable Models

Chirag Nagpal
Carnegie Mellon University
Pittsburgh, PA 15213
`chiragn@andrew.cmu.edu`

Yurun Tian
Carnegie Mellon University
Pittsburgh, PA 15213
`yurunt@andrew.cmu.edu`

Prabh Simran Baweja
Carnegie Mellon University
Pittsburgh, PA 15213
`pbaweja@andrew.cmu.edu`

Nataphop Tiamdao
Carnegie Mellon University
Pittsburgh, PA 15213
`ntiamdao@andrew.cmu.edu`

Michael Brzozowski
Carnegie Mellon University
Pittsburgh, PA 15213
`mbrzozow@andrew.cmu.edu`

1 Introduction

Variational Autoencoders (VAE) are one of the most popular unsupervised generative models that have achieved great performance for tasks such as image reconstruction and data compression. VAEs learn an embedding, the latent space, which captures the structure of a dataset in a more representative dimensionality and parameterization. This space can then be sampled for new points, which can be decoded from the embedding into new elements, such as images, in the original space. Standard VAEs rely on a singular Gaussian to model the distribution of elements in the latent space representation. Especially when it comes to datasets with multiple very distinct classes, this assumption becomes a limitation in the ability for the latent state to model the distribution of the entire dataset. To address this limitation, a VAE with Gaussian Mixture Model (GMM) is used to further structure the latent space as a mapping from the data points to a set of multiple Gaussian distributions, where the number of components, k, is a pre-defined hyperparameter. A naive guess for k is simply to use the number of classes in the dataset, but this is not always the optimal solution. Instead of arbitrarily hand-tuning this parameter to achieve higher flexibility of the latent space, we propose to extend the limited number of Gaussian distributions in the current GMM to an unlimited setting via the Hierarchical Infinite Gaussian Mixture Model. This will automate the learning process with no limit on the Bayesian prior, and bring higher generalization ability to the VAE.

2 Literature Review

2.1 Generative Architectures:VAE & GMVAE

Kingma, et al. (1995) [4] first introduced the VAE architecture. Dilokthanakul, et al. The objective of VAE could be formed as:

$$ELBO(\theta, \phi, \boldsymbol{x}) = E_{q_\phi(z|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x} \mid z)\right] - KL\left(q_\phi(z \mid \boldsymbol{x}) \| p(z)\right) \quad (1)$$

Because regular VAEs assume the latent variables to be isotropic Gaussian, learned representations are bound to a unimodal distribution. Under this assumption, it is challenging to learn complex latent representations. (2017) [3] expanded on the VAE architecture using Gaussian Mixture Models. The objective therefore changes into:

Preprint. Under review.

$$
\begin{aligned}
\mathrm{ELBO}(\theta, \boldsymbol{x}) = & E_{q_{\phi_z}(z|\boldsymbol{x})}\left[\log p_\theta(\boldsymbol{x} \mid z)\right] \\
& - E_{q_{\phi_w}(w|x) q_{\phi_z}(z|x)}\left[\sum_{k=1}^{K} p_\beta\left(y_k = 1 \mid \boldsymbol{w}, z\right) \log \frac{q_{\phi_z}(z \mid \boldsymbol{x})}{p_\beta\left(z \mid \mathbf{w}, y_k = 1\right)}\right] \\
& - KL\left(q_{\phi_w}(\boldsymbol{w} \mid \boldsymbol{x}) \| p(\boldsymbol{w})\right) \\
& - E_{q_{\phi_z}(z|x) q_{\phi_w}(w|x)}\left[KL\left(p_\beta(y \mid \boldsymbol{w}, \boldsymbol{z}) \| p(y)\right)\right]
\end{aligned} \tag{2}
$$

where the notations in both equations mean:

- Input data: $\boldsymbol{x} \sim p(\boldsymbol{x})$
- Latent variable $z : p(z) = \int p(z \mid \boldsymbol{w}, y) p(\boldsymbol{w}) p(y) d\boldsymbol{w} dy$
- Latent variable $\boldsymbol{w} : p(\boldsymbol{w}) = N(0, \boldsymbol{I})$
- Latent variable $y : y \sim \mathrm{Mult}(\pi)$ given that $\pi_i = \frac{1}{K}$

The Gaussian Mixture VAE model chooses the prior of the latent space as a mixture of Gaussians in order to overcome these limitations. The Gaussian Mixture Generative and Variational components are shown in Figure 1. The variables in the figures have the following meaning:

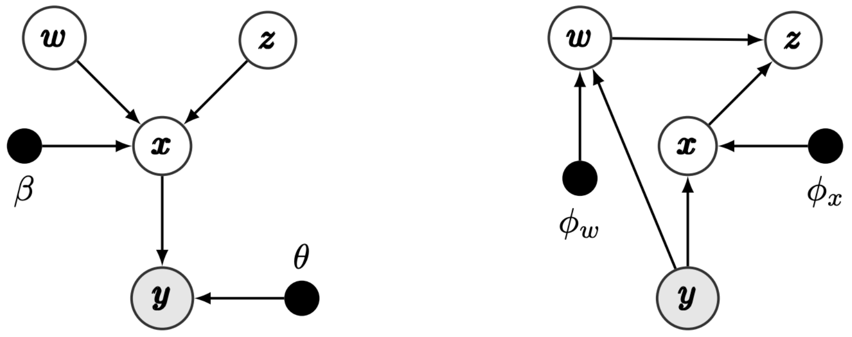

Figure 1: Gaussian Mixture VAEs generative model (left) and variational model (right). Credits: [3]

Gaussian Mixture VAEs provide competitive performance as compared to the state-of-the-art on popular datasets. Rasmussen (1999) [5] introduced the Infinite Gaussian Mixture Model. The Infinite Gaussian Mixture Model constructs a prior which does not assume the number of components Gaussians to be finite. In several applications, defining a limited number of components might not lead to an optimal representation of the data distribution. The Infinite Gaussian Mixture Model provides the model with flexibility to automatically determine the number of components.

2.2 Variational Inferecne: KMeans & GMM & DPGMM

The KMeans algorithm partitions data by trying to divide samples into groups of equal variance, thereby minimizing the inertia, or within-cluster sum of squares. The algorithm needs to define the number of clusters.

The Gaussian mixture model (GMM) is a probabilistic model that assumes all data points are generated from a mixture of **finite** Gaussian distributions with unknown parameters. The mixture model can be thought of as a generalization of k-sets of means to integrate information about the covariance structure of the data and possible Gaussian centers.

The Dirichlet operation is the prior probability distribution of sets with infinite partitions. Compared with the finite Gaussian mixture model, the covariance technique allows us to incorporate this prior structure into the Gaussian mixture model with almost no penalty at the time of inference.

The Dirichlet system is also applicable to the infinite mixing, but requires a large number of components to be used, and only represents the maximum concentration of the quantity and the maximum range of mixing.

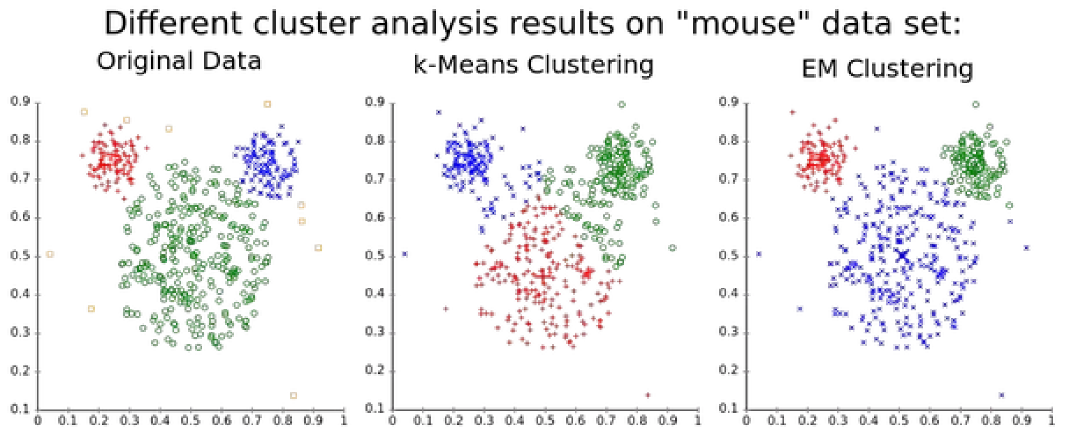

Figure 2: Difference between KMeans and EM-based clustering (GMMs) Credits: [2]

Figure 2 shows the difference between KMeans and Gaussian Mixture based models:

KMeans: look for a k that minimizes: $(x - \mu_k)^2$

Gaussian Mixture (EM clustering) : find k to minimize $\frac{(x-\mu_k)^2}{\sigma^2}$.

The difference (mathematically) is the denominator σ^2, which means GM takes variance into consideration when it calculates the measurement, whereas KMeans only calculates conventional Euclidean distance. In other words, KMeans calculate distance, while GM calculates **"weighted"** distance. [1]]

3 Proposed Network: GMVAE + DP-GMM Architecture

We propose an architecture that combines the Gaussian Mixture Variational Autoencoder (GMVAE) with the infinite mixture model (DP-GMM). The idea behind this approach is for the model to automatically calculate the number of components from the input data. We propose the problem as a **Maximization-Maximization** problem to approximate the Expectation Maximization (EM) algorithm.

The training method updates the weights of the means and variance in the GMVAE architecture after each epoch based on the results from the DP-GMM model. The latent dimension weights are fit using the DP-GMM model to get the resulting means and variances. The DP-GMM model will enable the network to learn the number of components in the dataset automatically. Furthermore, we experiment with both hard gumbel softmax method and soft gumbel softmax method. The training algorithm for our proposed network is as follows:

Algorithm 1 Proposed Training Algorithm

```
for epoch do
    $X_{latent} \leftarrow Encoder(X)$
    $mean, var \leftarrow DPGMM(X_{latent})$
    $z \leftarrow reparameterize(X_{latent}, mean, var)$
    $\hat{X} \leftarrow Decoder(z)$
    $loss \leftarrow diverge(X, \hat{X})$
    $loss.backward$
end for
```

4 Experiments

4.1 Dataset

We evaluate our models on the MNIST dataset. The MNIST dataset contains handwritten images including 60,000 training examples and 10,000 test examples. We used this dataset because it contains high dimensional nonlinear data with clear class partitions. It serves as a good starting point compared to other complicated datasets, which makes it easier for sanity checking of our novel techniques.

4.2 Evaluation Criterion

Completeness Score (CS)

Completeness metric of a cluster labeling given a ground truth. A clustering result satisfies completeness if all the data points that are members of a given class are elements of the same cluster. The score is score between 0.0 and 1.0. 1.0 stands for perfectly complete labeling. For example, if the ground truth class labels are $[0, 0, 1, 1]$, and the cluster IDs assigned by the clustering method are $[1, 1, 0, 0]$, the score is 1.0.

Normalized mutual information (NMI)

Normalized Mutual Information between two clusterings. Normalized Mutual Information (NMI) is a normalization of the Mutual Information (MI) score to scale the results between 0.0 (no mutual information) and 1.0 (perfect correlation)

$$NMI(Y, \Omega) \equiv \frac{MI(Y, \Omega)}{(H(Y) + H(\Omega))/2}$$

where $Y = \{y_1, y_2, \cdots y_K\}$ is the true-label based partition of the test dataset and $\Omega = \{\omega_1, \omega_2, \cdots \omega_M\}$ is the clustering method partition of the test set. H is the entropy of a random variable.

The metrics CS and NMI are measurements for variational inference. In our experiments, they demonstrate the feature quality in the latent space as well as the clustering quality.

4.3 Baseline Models

As outlined in Table 1, the two base networks we use are VAE and GMVAE. We extend each with a fitting of the latent space using different models. We include fixed VAE, VAE with K-means, VAE with GMM, VAE with DP-GMM, GMVAE, and GMVAE with DP-GMM.

Table 1: Network and Model Configurations

Network	Model	Latent-dims	Components
VAE	Fixed	{8, 10, 16, 64}	1
	K-means	{8, 10, 16, 64}	10
	GMM	{8, 10, 16, 64}	10
	DP-GMM	{8, 10, 16, 64}	Infinite
GM-VAE	Fixed	{8, 10, 16, 64}	10
	DP-GMM	{8, 10, 16, 64}	Infinite

Baseline evaluation is performed on MNIST. We evaluate the models outlined above on the evaluation metrics to form our baseline suite. In addition to the metrics, we display the reconstruction results from sampling the latent spaces in each of the different model results.

4.4 Proposed Network Experiments

We performed the following experiments with our proposed network:

- **Unified DPGMM and GMVAE using Algorithm 1 with soft gumbel softmax**: Algorithm 1 is used along with soft gumbel softmax to train the model for 100 epochs for latent dimension 8, 10, 16, 64.
- **Unified DPGMM and GMVAE using Algorithm 1 with hard gumbel softmax**: Algorithm 1 is used along with hard gumbel softmax to train the model for 100 epochs for latent dimension 8, 10, 16, 64.
- **Unified DPGMM and GMVAE using Algorithm 1 with pretrained model for GMVAE architecture**: Weights from the network trained only with the GMVAE architecture for 70 epochs are loaded to the GMVAE + DP-GMM model.

4.5 Implementation

The code for all the models can be found *here*.

5 Results

As shown in table 2, over all the methods and all values of $\mathbf{Z}$, VAE+GMM gives the best results when $\mathbf{Z} = 16$ for both CS and NMI scores. DP-GMM displays better performance with relatively small $\mathbf{Z}$ values. When $\mathbf{Z}$ is large, GMM related methods (VAE+GMM, GMVAE) demonstrate priorities over other methods. This provides us the insight that DP-GMM may generalize better with a small latent dimension rather than a large latent dimension. Notably, $\mathbf{Z} = 64$ doesn't give the best score among all the other values, which indicates that using the largest latent dimension does not necessarily provide the most informative and interpretable features.

The results for our proposed network suggest that the training procedure is not stable and the loss, accuracy and NMI of the models are fluctuating every epoch. Figure 3 shows the metrics for the model trained for 100 epochs for all three proposed network experiments.

Our proposed model is undercomplete because the training is unstable and as a result the alternating training scheme is not reaching the baseline of GMVAE + DPGMM. One potential reason for this could be the difficulty of coordinating the two separate modules. When we perform alternating training between GMVAE and DPGMM, they are optimizing their own objectives and thus do not communicate well with each other. Additional experiments with hard gumbel softmax instead of soft gumbel softmax, and using a pretrained GMVAE network were not able to remedy these issues.

The hard gumbel softmax experiment and pretrained GMVAE network experiment do not seem to have stable training procedures as well. This suggests that additional modifications within the training procedure along with developing a unified single neural module will be required to approach better results and more stable training in the future.

Table 2: Results

	CS	NMI	CS	NMI	CS	NMI	CS	NMI
Model	**Z = 8**		**Z = 10**		**Z = 16**		**Z = 64**	
VAE+K-Means	0.61	0.61	0.64	0.64	0.63	0.63	0.61	0.61
VAE+GMM	0.77	0.77	0.82	0.82	**0.84**	**0.84**	0.67	0.66
VAE+DP-GMM	0.77	0.76	**0.83**	**0.83**	0.78	0.77	0.70	0.70
GMVAE	0.67	0.69	0.78	0.74	0.73	0.71	**0.83**	**0.82**
GMVAE+DP-GMM	**0.80**	**0.79**	0.80	0.79	0.72	0.71	0.81	0.80
Proposed Network (Soft Gumbel)	0.77	0.65	0.76	0.68	0.68	0.58	0.70	0.64
Proposed Network (Hard Gumbel)	0.69	0.63	0.73	0.63	0.67	0.57	0.60	0.53

5.1 Reconstruction

As demonstrated in figures 4 and 5 the GMVAE model and the VAE model are able to accurately reconstruct the images using the latent dimension, z=64.

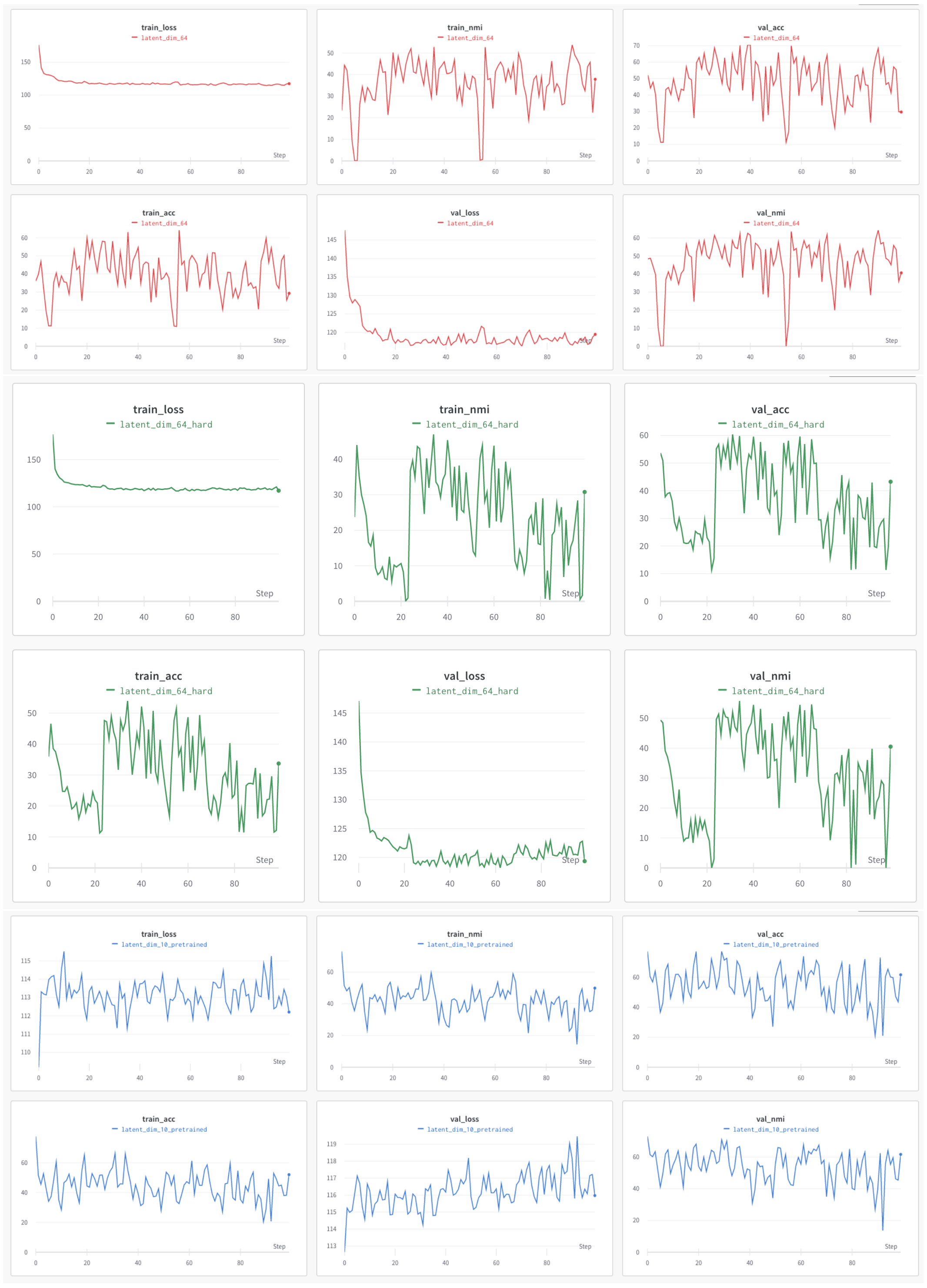

Figure 3: Training, Validation loss, accuracy and NMI score for the experiments. The red color graphs are for the experiment with latent dimension 64 and soft gumbel softmax. The green color graphs are the experiment with latent dimension 64 and hard gumbel softmax. The blue color graphs are for the experiment with latent dimension 10 along with a pretrained model. The results show that the training procedure is not stable.

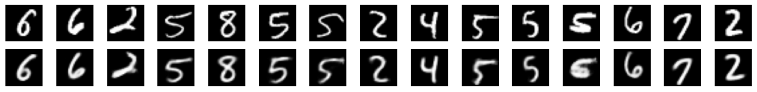

Figure 4: Reconstruction of randomly sampled MNIST images using the GMVAE model.

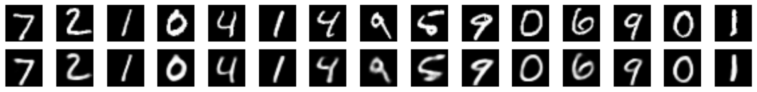

Figure 5: Reconstruction of randomly sampled MNIST images using the VAE model.

5.2 Latent Dimension Scatter Plots

We draw scatter plots for all the latent dimensions of the various models. We analyze the scatter plots for the VAE model with latent dimension, z=64, based on multiple techniques. Figure 6 shows the scatter plots for all of the methods. We see that for the VAE model, all of the components are near each other and they are all close to the center.

When we analyze the results for the GMVAE models, we see that the components are more spread out, and clearly separated in different clusters. They are further away from the center as well in this case. Figure 7 displays the results for the GMVAE model and the DP-GMM model fit on top of GMVAE with latent dimension, z=64.

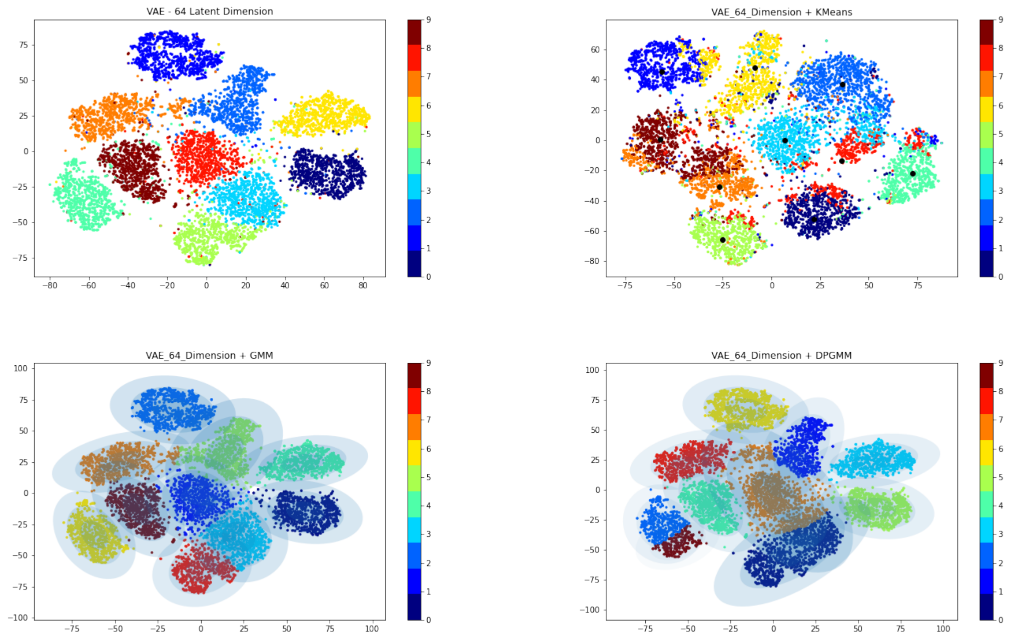

Figure 6: Scatter Plots: VAE with latent dimension (z=64) (top-left), VAE+K-means with latent dimension (z=64) (top-right), VAE+GMM with latent dimension (z=64) (bottom-left), VAE+DP-GMM with latent dimension (z=64) (bottom-right).

5.3 Generation

We further analyze the generation properties of the GMVAE model. As we see above with all of the other experiments, latent dimension, z=64 provides us with the best generation results as well. Figure

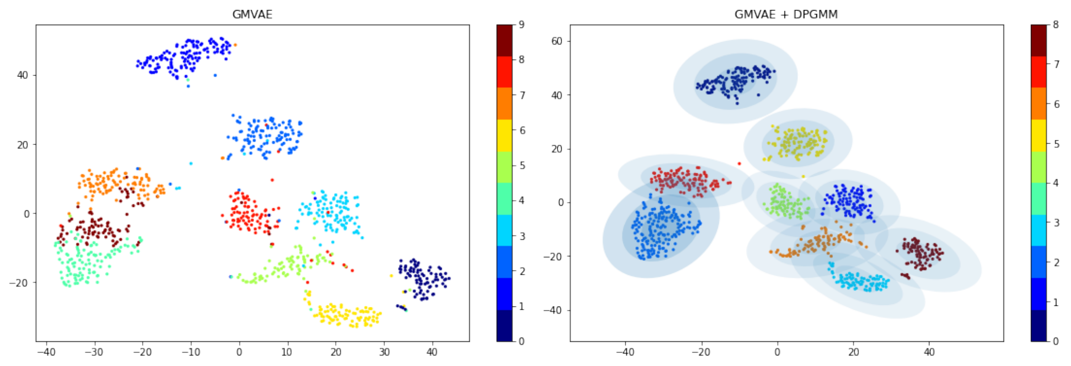

Figure 7: Scatter Plot for the latent dimension (z=64) for GMVAE model (left) and DP-GMM fit on top of GMVAE latent features (right).

8 demonstrates the results from 10 images randomly sampled from each component of the GMVAE model. We can see that for most of the classes, the model has been able to learn the digit accurately. We also notice that the model is unable to clearly distinguish between the digits 4, 9 and 7. This behaviour is visible from the scatter plots of the GMVAE models as well (Figure 7).

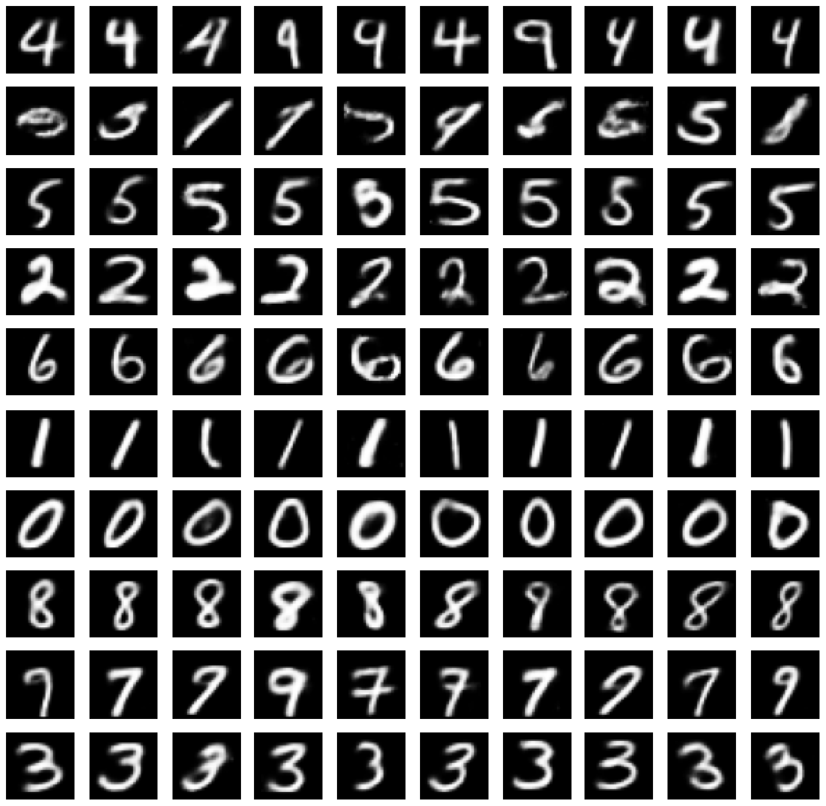

Figure 8: Generation of 10 images randomly sampled from each component of GMVAE model.

6 Conclusion & Future work

Overall, no single method dominates all the best results. We believe one of the reasons for this is that different latent dimension results in different representations of the latent dimension space.

Additionally, we performed random hyperparameter search which is faster and proves to be efficient for deep learning tasks. However, thorough grid search of hyperparameters could potentially yield a more optimal environment.

Training is not stable for our proposed method of GMVAE+DP-GMM (unified) due to difficulty faced in coordinating both models. When When alternating training between GMVAE and DPGMM, they are optimized based on their own objective and thus cannot communicate well with each other. We believe this to be the root of the problem based on similar issues encountered in other multi-module training approaches. Examples are the difficulties faced in training the generator and discriminator together in GANs, or syncing two biLSTMs together in ELMo. The most promising solution is to develop a unified single neural module to conduct the training.

We are considering extending the work to create a unified model which realizes the two modules cohesively. Another consideration is that we provided 10 components into the DPGMM algorithm at the current stage for sanity check. In this case, the model can choose not to use all of the components by setting some component weights to values very close to zero. So far we have not observe these 0 weights, which indicates means the model is aware that 10 is the answer. In the future, we would provide higher numbers of components as next steps. In addition, we can explore different ways of formulating the problem. Currently we adopt the Maximize-Maximize paradigm which has proven to be sub-optimal, and can instead implement the Expectation-Maximization paradigm.

References

[1] K-means vs mixture model gaussian. `https://www.quora.com/What-is-the-difference-between-K-means-and-the-mixture-model-of-Gaussian`. Accessed: 2010-09-30.

[2] Wikipedia: Expectation maximization. `https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm`. Accessed: 2010-09-30.

[3] Nat Dilokthanakul, Pedro A. M. Mediano, Marta Garnelo, Matthew C. H. Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. Deep unsupervised clustering with gaussian mixture variational autoencoders. *CoRR*, abs/1611.02648, 2016.

[4] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[5] Carl Rasmussen. The infinite gaussian mixture model. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 2000.