DEEP UNSUPERVISED CLUSTERING WITH GAUSSIAN MIXTURE VARIATIONAL AUTOENCODERS

Nat Dilokthanakul[1,*], **Pedro A. M. Mediano**[1], **Marta Garnelo**[1],
Matthew C. H. Lee[1], **Hugh Salimbeni**[1], **Kai Arulkumaran**[2] **& Murray Shanahan**[1]
[1]Department of Computing, [2]Department of Bioengineering
Imperial College London
London, UK
*n.dilokthanakul14@imperial.ac.uk

ABSTRACT

We study a variant of the variational autoencoder model (VAE) with a Gaussian mixture as a prior distribution, with the goal of performing unsupervised clustering through deep generative models. We observe that the known problem of over-regularisation that has been shown to arise in regular VAEs also manifests itself in our model and leads to cluster degeneracy. We show that a heuristic called minimum information constraint that has been shown to mitigate this effect in VAEs can also be applied to improve unsupervised clustering performance with our model. Furthermore we analyse the effect of this heuristic and provide an intuition of the various processes with the help of visualizations. Finally, we demonstrate the performance of our model on synthetic data, MNIST and SVHN, showing that the obtained clusters are distinct, interpretable and result in achieving competitive performance on unsupervised clustering to the state-of-the-art results.

1 INTRODUCTION

Unsupervised clustering remains a fundamental challenge in machine learning research. While long-established methods such as k-means and Gaussian mixture models (GMMs) (Bishop, 2006) still lie at the core of numerous applications (Aggarwal & Reddy, 2013), their similarity measures are limited to local relations in the data space and are thus unable to capture hidden, hierarchical dependencies in latent spaces. Alternatively, deep generative models can encode rich latent structures. While they are not often applied *directly* to unsupervised clustering problems, they can be used for dimensionality reduction, with classical clustering techniques applied to the resulting low-dimensional space (Xie et al., 2015). This is an unsatisfactory approach as the assumptions underlying the dimensionality reduction techniques are generally independent of the assumptions of the clustering techniques.

Deep generative models try to estimate the density of observed data under some assumptions about its latent structure, i.e., its hidden causes. They allow us to reason about data in more complex ways than in models trained purely through supervised learning. However, inference in models with complicated latent structures can be difficult. Recent breakthroughs in approximate inference have provided tools for constructing tractable inference algorithms. As a result of combining differentiable models with variational inference, it is possible to scale up inference to datasets of sizes that would not have been possible with earlier inference methods (Rezende et al., 2014). One popular algorithm under this framework is the variational autoencoder (VAE) (Kingma & Welling, 2013; Rezende et al., 2014).

In this paper, we propose an algorithm to perform unsupervised clustering within the VAE framework. To do so, we postulate that generative models can be tuned for unsupervised clustering by making the assumption that the observed data is generated from a multimodal prior distribution, and, correspondingly, construct an inference model that can be directly optimised using the reparameterization trick. We also show that the problem of over-regularisation in VAEs can severely effect the performance of clustering, and that it can be mitigated with the minimum information constraint introduced by Kingma et al. (2016).

1.1 Related Work

Unsupervised clustering can be considered a subset of the problem of disentangling latent variables, which aims to find structure in the latent space in an unsupervised manner. Recent efforts have moved towards training models with disentangled latent variables corresponding to different factors of variation in the data. Inspired by the learning pressure in the ventral visual stream, Higgins et al. (2016) were able to extract disentangled features from images by adding a regularisation coefficient to the lower bound of the VAE. As with VAEs, there is also effort going into obtaining disentangled features from generative adversarial networks (GANs) (Goodfellow et al., 2014). This has been recently achieved with InfoGANs (Chen et al., 2016a), where structured latent variables are included as part of the noise vector, and the mutual information between these latent variables and the generator distribution is then maximised as a mini-max game between the two networks. Similarly, Tagger (Greff et al., 2016), which combines iterative amortized grouping and ladder networks, aims to perceptually group objects in images by iteratively denoising its inputs and assigning parts of the reconstruction to different groups. Johnson et al. (2016) introduced a way to combine amortized inference with stochastic variational inference in an algorithm called structured VAEs. Structured VAEs are capable of training deep models with GMM as prior distribution. Shu et al. (2016) introduced a VAE with a multimodal prior where they optimize the variational approximation to the standard variational objective showing its performance in video prediction task.

The work that is most closely related to ours is the stacked generative semi-supervised model (M1+M2) by Kingma et al. (2014). One of the main differences is the fact that their prior distribution is a neural network transformation of both continuous and discrete variables, with Gaussian and categorical priors respectively. The prior for our model, on the other hand, is a neural network transformation of Gaussian variables, which parametrise the means and variances of a mixture of Gaussians, with categorical variables for the mixture components. Crucially, Kingma et al. (2014) apply their model to semi-supervised classification tasks, whereas we focus on unsupervised clustering. Therefore, our inference algorithm is more specific to the latter.

We compare our results against several orthogonal state-of-the-art techniques in unsupervised clustering with deep generative models: deep embedded clustering (DEC) (Xie et al., 2015), adversarial autoencoders (AAEs) (Makhzani et al., 2015) and categorial GANs (CatGANs) (Springenberg, 2015).

2 Variational Autoencoders

VAEs are the result of combining variational Bayesian methods with the flexibility and scalability provided by neural networks (Kingma & Welling, 2013; Rezende et al., 2014). Using variational inference it is possible to turn intractable inference problems into optimisation problems (Wainwright & Jordan, 2008), and thus expand the set of available tools for inference to include optimisation techniques as well. Despite this, a key limitation of classical variational inference is the need for the likelihood and the prior to be conjugate in order for most problems to be tractably optimised, which in turn can limit the applicability of such algorithms. Variational autoencoders introduce the use of neural networks to output the conditional posterior (Kingma & Welling, 2013) and thus allow the variational inference objective to be tractably optimised via stochastic gradient descent and standard backpropagation. This technique, known as the reparametrisation trick, was proposed to enable backpropagation through continuous stochastic variables. While under normal circumstances backpropagation through stochastic variables would not be possible without Monte Carlo methods, this is bypassed by constructing the latent variables through the combination of a deterministic function and a separate source of noise. We refer the reader to Kingma & Welling (2013) for more details.

3 Gaussian Mixture Variational Autoencoders

In regular VAEs, the prior over the latent variables is commonly an isotropic Gaussian. This choice of prior causes each dimension of the multivariate Gaussian to be pushed towards learning a separate continuous factor of variation from the data, which can result in learned representations that are structured and disentangled. While this allows for more interpretable latent variables (Higgins et al., 2016), the Gaussian prior is limited because the learnt representation can only be unimodal and does not allow for more complex representations. As a result, numerous extensions to the VAE have been developed, where more complicated latent representations can be learned by specifying increasingly complex priors (Chung et al., 2015; Gregor et al., 2015; Eslami et al., 2016).

In this paper we choose a mixture of Gaussians as our prior, as it is an intuitive extension of the unimodal Gaussian prior. If we assume that the observed data is generated from a mixture of Gaussians, inferring the class of a data point is equivalent to inferring which mode of the latent distribution the data point was generated from. While this gives us the possibility to segregate our latent space into distinct classes, inference in this model is non-trivial. It is well known that the reparametrisation trick which is generally used for VAEs cannot be directly applied to discrete variables. Several possibilities for estimating the gradient of discrete variables have been proposed (Glynn, 1990; Titsias & Lázaro-Gredilla, 2015). Graves (2016) also suggested an algorithm for backpropagation through GMMs. Instead, we show that by adjusting the architecture of the standard VAE, our estimator of the variational lower bound of our Gaussian mixture variational autoencoder (GMVAE) can be optimised with standard backpropagation through the reparametrisation trick, thus keeping the inference model simple.

3.1 Generative and Recognition Models

Consider the generative model $p_{\beta,\theta}(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z}) = p(\boldsymbol{w})p(\boldsymbol{z})p_{\beta}(\boldsymbol{x}|\boldsymbol{w}, \boldsymbol{z})p_{\theta}(\boldsymbol{y}|\boldsymbol{x})$, where an observed sample $\boldsymbol{y}$ is generated from a set of latent variables $\boldsymbol{x}$, $\boldsymbol{w}$ and $\boldsymbol{z}$ under the following process:

$$\boldsymbol{w} \sim \mathcal{N}(0, \boldsymbol{I}) \tag{1a}$$

$$\boldsymbol{z} \sim Mult(\boldsymbol{\pi}) \tag{1b}$$

$$\boldsymbol{x}|\boldsymbol{z}, \boldsymbol{w} \sim \prod_{k=1}^{K} \mathcal{N}\left(\boldsymbol{\mu}_{z_k}(\boldsymbol{w}; \beta), diag\left(\boldsymbol{\sigma}^2_{z_k}(\boldsymbol{w}; \beta)\right)\right)^{z_k} \tag{1c}$$

$$\boldsymbol{y}|\boldsymbol{x} \sim \mathcal{N}\left(\boldsymbol{\mu}(\boldsymbol{x}; \theta), diag\left(\boldsymbol{\sigma}^2(\boldsymbol{x}; \theta)\right)\right) \text{ or } \mathcal{B}(\boldsymbol{\mu}(\boldsymbol{x}; \theta)) \,. \tag{1d}$$

where K is a predefined number of components in the mixture, and $\boldsymbol{\mu}_{z_k}(\cdot; \beta), \boldsymbol{\sigma}^2_{z_k}(\cdot; \beta), \boldsymbol{\mu}(\cdot; \theta)$, and $\boldsymbol{\sigma}^2(\cdot; \theta)$ are given by neural networks with parameters β and θ, respectively. That is, the observed sample $\boldsymbol{y}$ is generated from a neural network observation model parametrised by θ and the continuous latent variable $\boldsymbol{x}$. Furthermore, the distribution of $\boldsymbol{x}|\boldsymbol{w}$ is a Gaussian mixture with means and variances specified by another neural network model parametrised by β and with input $\boldsymbol{w}$.

More specifically, the neural network parameterised by β outputs a set of K means $\boldsymbol{\mu}_{z_k}$ and K variances $\boldsymbol{\sigma}^2_{z_k}$, given $\boldsymbol{w}$ as input. A one-hot vector $\boldsymbol{z}$ is sampled from the mixing probability $\boldsymbol{\pi}$, which chooses one component from the Gaussian mixture. We set the parameter $\pi_k = K^{-1}$ to make $\boldsymbol{z}$ uniformly distributed. The generative and variational views of this model are depicted in Fig. 1.

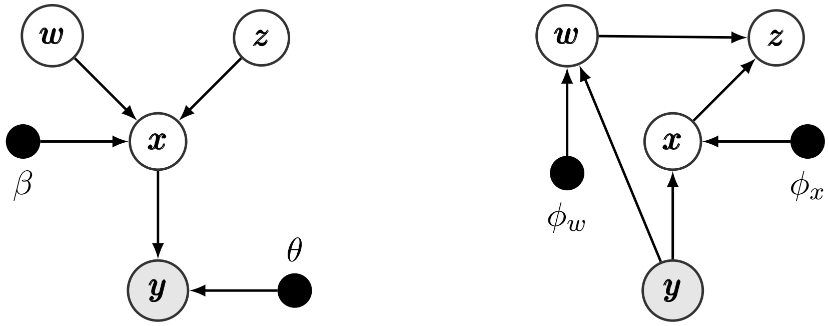


Figure 1: Graphical models for the Gaussian mixture variational autoencoder (GMVAE) showing the generative model (left) and the variational family (right).

3.2 Inference with the Recognition Model

The generative model is trained with the variational inference objective, i.e. the log-evidence lower bound (ELBO), which can be written as

$$\mathcal{L}_{ELBO} = \mathbb{E}_q \left[\frac{p_{\beta,\theta}(\boldsymbol{y}, \boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z})}{q(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z}|\boldsymbol{y})} \right]. \tag{2}$$

We assume the mean-field variational family $q(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z}|\boldsymbol{y})$ as a proxy to the posterior which factorises as $q(\boldsymbol{x}, \boldsymbol{w}, \boldsymbol{z}|\boldsymbol{y}) = \prod_i q_{\phi_x}(\boldsymbol{x}_i|\boldsymbol{y}_i) q_{\phi_w}(\boldsymbol{w}_i|\boldsymbol{y}_i) p_\beta(\boldsymbol{z}_i|\boldsymbol{x}_i, \boldsymbol{w}_i)$, where i indexes over data points. To simplify further notation, we will drop i and consider one data point at a time. We parametrise each variational factor with the recognition networks ϕ_x and ϕ_w that output the parameters of the variational distributions and specify their form to be Gaussian posteriors. We derived the z-posterior, $p_\beta(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{w})$, as:

$$\begin{aligned} p_\beta(z_j = 1|\boldsymbol{x}, \boldsymbol{w}) &= \frac{p(z_j = 1) p(\boldsymbol{x}|z_j = 1, \boldsymbol{w})}{\sum_{k=1}^{K} p(z_k = 1) p(\boldsymbol{x}|z_j = 1, \boldsymbol{w})} \\ &= \frac{\pi_j \mathcal{N}(\boldsymbol{x}|\mu_j(\boldsymbol{w}; \beta), \sigma_j(\boldsymbol{w}; \beta))}{\sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{x}|\mu_k(\boldsymbol{w}; \beta), \sigma_k(\boldsymbol{w}; \beta))} . \end{aligned} \tag{3}$$

The lower bound can then be written as,

$$\begin{aligned} \mathcal{L}_{ELBO} = \mathbb{E}_{q(\boldsymbol{x}|\boldsymbol{y})} \big[\log p_\theta(\boldsymbol{y}|\boldsymbol{x}) \big] &- \mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{y}) p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{w})} \big[KL(q_{\phi_x}(\boldsymbol{x}|\boldsymbol{y}) || p_\beta(\boldsymbol{x}|\boldsymbol{w}, \boldsymbol{z})) \big] \\ - KL(q_{\phi_w}(\boldsymbol{w}|\boldsymbol{y}) || p(\boldsymbol{w})) &- \mathbb{E}_{q(\boldsymbol{x}|\boldsymbol{y}) q(\boldsymbol{w}|\boldsymbol{y})} \big[KL(p_\beta(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{w}) || p(\boldsymbol{z})) \big]. \end{aligned} \tag{4}$$

We refer to the terms in the lower bound as the reconstruction term, conditional prior term, w-prior term and z-prior term respectively.

3.2.1 The Conditional Prior Term

The reconstruction term can be estimated by drawing Monte Carlo samples from $q(\boldsymbol{x}|\boldsymbol{y})$, where the gradient can be backpropagated with the standard reparameterisation trick (Kingma & Welling, 2013). The w-prior term can be calculated analytically.

Importantly, by constructing the model this way, the conditional prior term can be estimated using Eqn. 5 without the need to sample from the discrete distribution $p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{w})$.

$$\begin{gathered} \mathbb{E}_{q(\boldsymbol{w}|\boldsymbol{y}) p(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{w})} \Big[KL\big(q_{\phi_x}(\boldsymbol{x}|\boldsymbol{y}) || p_\beta(\boldsymbol{x}|\boldsymbol{w}, \boldsymbol{z})\big) \Big] \approx \\ \frac{1}{M} \sum_{j=1}^{M} \sum_{k=1}^{K} p_\beta(z_k = 1|\boldsymbol{x}^{(j)}, \boldsymbol{w}^{(j)}) KL \Big(q_{\phi_x}(\boldsymbol{x}|\boldsymbol{y}) || p_\beta(\boldsymbol{x}|\boldsymbol{w}^{(j)}, z_k = 1) \Big) \end{gathered} \tag{5}$$

Since $p_\beta(\boldsymbol{z}|\boldsymbol{x}, \boldsymbol{w})$ can be computed for all $\boldsymbol{z}$ with one forward pass, the expectation over it can be calculated in a straightforward manner and backpropagated as usual. The expectation over $q_{\phi_w}(\boldsymbol{w}|\boldsymbol{y})$ can be estimated with M Monte Carlo samples and the gradients can be backpropagated via the reparameterisation trick. This method of calculating the expectation is similar to the marginalisation approach of Kingma et al. (2014), with a subtle difference. Kingma et al. (2014) need multiple forward passes to obtain each component of the z-posterior. Our method requires wider output layers of the neural network parameterised by β, but only need one forward pass. Both methods scale up linearly with the number of clusters.

3.3 The KL Cost of the Discrete Latent Variable

The most unusual term in our ELBO is the z-prior term. The z-posterior calculates the clustering assignment probability directly from the value of x and w, by asking how far x is from each of the cluster positions generated by w. Therefore, the z-prior term can reduce the KL divergence between the z-posterior and the uniform prior by concurrently manipulating the position of the clusters and the encoded point x. Intuitively, it would try to merge the clusters by maximising the overlap between them, and moving the means closer together. This term, similar to other KL-regularisation terms, is in tension with the reconstruction term, and is expected to be over-powered as the amount of training data increases.

3.4 The Over-regularisation Problem

The possible overpowering effect of the regularisation term on VAE training has been described numerous times in the VAE literature (Bowman et al., 2015; Sønderby et al., 2016; Kingma et al., 2016; Chen et al., 2016b). As a result of the strong influence of the prior, the obtained latent representations are often overly simplified and poorly represent the underlying structure of the data. So far there have been two main approaches to overcome this effect: one solution is to anneal the KL term during training by allowing the reconstruction term to train the autoencoder network before slowly incorporating the regularization from the KL term (Sønderby et al., 2016). The other main approach involves modifying the objective function by setting a cut-off value that removes the effect of the KL term when it is below a certain threshold (Kingma et al., 2016). As we show in the experimental section below, this problem of over-regularisation is also prevalent in the assignment of the GMVAE clusters and manifests itself in large degenerate clusters. While we show that the second approach suggested by Kingma et al. (2016) does indeed alleviate this merging phenomenon, finding solutions to the over-regularization problem remains a challenging open problem.

4 Experiments

The main objective of our experiments is not only to evaluate the accuracy of our proposed model, but also to understand the optimisation dynamics involved in the construction of meaningful, differentiated latent representations of the data. This section is divided in three parts:

1. We first study the inference process in a low-dimensional synthetic dataset, and focus in particular on how the over-regularisation problem affects the clustering performance of the GMVAE and how to alleviate the problem;
2. We then evaluate our model on an MNIST unsupervised clustering task; and
3. We finally show generated images from our model, conditioned on different values of the latent variables, which illustrate that the GMVAE can learn disentangled, interpretable latent representations.

Throughout this section we make use of the following datasets:

- **Synthetic data**: We create a synthetic dataset mimicking the presentation of Johnson et al. (2016), which is a 2D dataset with 10,000 data points created from the arcs of 5 circles.
- **MNIST**: The standard handwritten digits dataset, composed of 28x28 grayscale images and consisting of 60,000 training samples and 10,000 testing samples (LeCun et al., 1998).
- **SVHN**: A collection of 32x32 images of house numbers (Netzer et al., 2011). We use the cropped version of the standard and the extra training sets, adding up to a total of approximately 600,000 images.

4.1 Synthetic data

We quantify clustering performance by plotting the magnitude of the z-prior term described in Eqn. 6 during training. This quantity can be thought of as a measure of how much different clusters overlap. Since our goal is to achieve meaningful clustering in the latent space, we would expect this quantity to go down as the model learns the separate clusters.

$$\mathcal{L}_z = -\mathbb{E}_{q(\boldsymbol{x}|\boldsymbol{y})q(\boldsymbol{w}|\boldsymbol{y})}\left[KL(p_\beta(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{w})||p(\boldsymbol{z}))\right] \tag{6}$$

Empirically, however, we have found this not to be the case. The latent representations that our model converges to merges all classes into the same large cluster instead of representing information about the different clusters, as can be seen in Figs. 2d and 3a. As a result, each data point is equally likely to belong to any of clusters, rendering our latent representations completely uninformative with respect to the class structure.

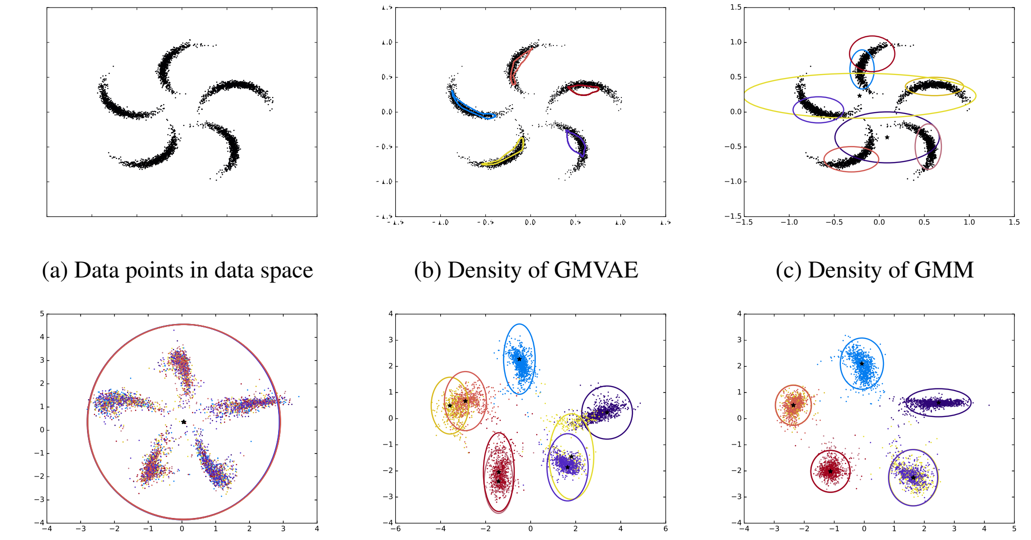
We argue that this phenomenon can be interpreted as the result of over-regularisation by the z-prior term. Given that this quantity is driven up by the optimisation of KL term in the lower bound, it reaches its maximum possible value of zero, as opposed to decreasing with training to ensure encoding of information about the classes. We suspect that the prior has too strong of an influence in the initial training phase and drives the model parameters into a poor local optimum that is hard to be driven out off by the reconstruction term later on.

This observation is conceptually very similar to the over-regularisation problem encountered in regular VAEs and we thus hypothesize that applying similar heuristics should help alleviate the problem. We show in Fig. 2f that by using the previously mentioned modification to the lower-bound proposed by Kingma et al. (2016), we can avoid the over-regularisation caused by the z-prior. This is achieved by maintaining the cost from the z-prior at a constant value λ until it exceeds that threshold. Formally, the modified z-prior term is written as:

$$\mathcal{L}'_z = -\max(\lambda, \mathbb{E}_{q(\boldsymbol{x}|\boldsymbol{y})q(\boldsymbol{w}|\boldsymbol{y})}\left[KL(p_\beta(\boldsymbol{z}|\boldsymbol{x},\boldsymbol{w})||p(\boldsymbol{z}))\right]) \tag{7}$$

This modification suppresses the initial effect of the z-prior to merge all clusters thus allowing them to spread out until the cost from the z-prior cost is high enough. At that point its effect is significantly reduced and is mostly limited to merging individual clusters that are overlapping sufficiently. This can be seen clearly in Figs. 2e and 2f. The former shows the clusters before the z-prior cost is taken into consideration, and as such the clusters have been able to spread out. Once the z-prior is activated, clusters that are very close together will be merged as seen in Fig. 2f.

Finally, in order to illustrate the benefits of using neural networks for the transformation of the distributions, we compare the density observed by our model (Fig. 2c) with a regular GMM (Fig. 2c) in data space. As illustrated by the figures, the GMVAE allows for a much richer, and thus more accurate representations than regular GMMs, and is therefore more successful at modelling non-Gaussian data.



(a) Data points in data space

(b) Density of GMVAE

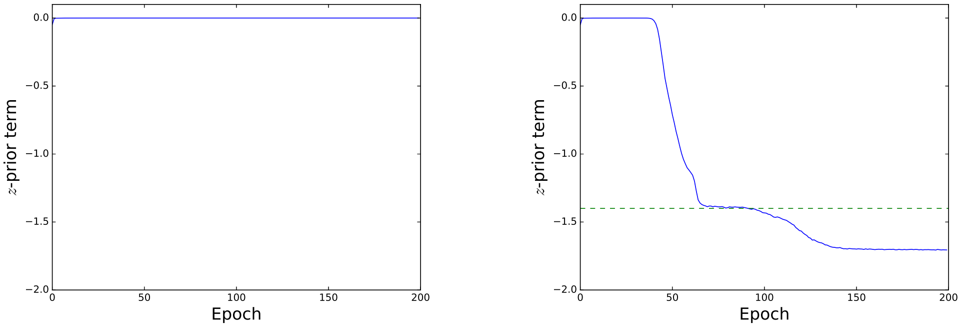
(c) Density of GMM

(d) Latent space, at poor optimum

(e) Latent space, clusters spreading

(f) Latent space, at convergence

Figure 2: **Visualisation of the synthetic dataset**: (a) Data is distributed with 5 modes on the 2 dimensional data space. (b) GMVAE learns the density model that can model data using a mixture of non-Gaussian distributions in the data space. (c) GMM cannot represent the data as well because of the restrictive Gaussian assumption. (d) GMVAE, however, suffers from over-regularisation and can result in poor minima when looking at the latent space. (e) Using the modification to the ELBO (Kingma et al., 2016) allows the clusters to spread out. (f) As the model converges the z-prior term is activated and regularises the clusters in the final stage by merging excessive clusters.



(a) z-prior term with normal ELBO

(b) z-prior term with the modification

Figure 3: **Plot of z-prior term**: (a) Without information constraint, GMVAE suffers from over-regularisation as it converges to a poor optimum that merges all clusters together to avoid the KL cost. (b) Before reaching the threshold value (dotted line), the gradient from the z-prior term can be turned off to avoid the clusters from being pulled together (see text for details). By the time the threshold value is reached, the clusters are sufficiently separated. At this point the activated gradient from the z-prior term only merges very overlapping clusters together. Even after activating its gradient the value of the z-prior continues to decrease as it is over-powered by other terms that lead to meaningful clusters and better optimum.

4.2 Unsupervised Image Clustering

We now assess the model's ability to represent discrete information present in the data on an image clustering task. We train a GMVAE on the MNIST training dataset and evaluate its clustering performance on the test dataset. To compare the cluster assignments given by the GMVAE with the true image labels we follow the evaluation protocol of Makhzani et al. (2015), which we summarise here for clarity. In this method, we find the element of the test set with the highest probability of belonging to cluster i and assign that label to all other test samples belonging to i. This is then repeated for all clusters $i = 1, ..., K$, and the assigned labels are compared with the true labels to obtain an unsupervised classification error rate.

While we observe the cluster degeneracy problem when training the GMVAE on the synthetic dataset, the problem does not arise with the MNIST dataset. We thus optimise the GMVAE using the ELBO directly, without the need for any modifications. A summary of the results obtained on the MNIST benchmark with the GMVAE as well as other recent methods is shown in Table 1. We achieve classification scores that are competitive with the state-of-the-art techniques[1], except for adversarial autoencoders (AAE). We suspect the reason for this is, again, related to the KL terms in the VAE's objective. As indicated by Hoffman et al., the key difference in the adversarial autoencoders objective is the replacement of the KL term in the ELBO by an adversarial loss that allows the latent space to be manipulated more carefully (Hoffman & Johnson, 2016). Details of the network architecture used in these experiments can be found in Appendix A.

Empirically, we observe that increasing the number of Monte Carlo samples and the number of clusters makes the GMVAE more robust to initialisation and more stable as shown in Fig. 4. If fewer samples or clusters are used then the GMVAE can occasionally converge faster to poor local minima, missing some of the modes of the data distribution.

[1] It is worth noting that shortly after our initial submission, Rui Shu published a blog post (http://ruishu.io/2016/12/25/gmvae/) with an analysis on Gaussian mixture VAEs. In addition to providing insightful comparisons to the aforementioned M2 algorithm, he implements a version that achieves competitive clustering scores using a comparably simple network architecture. Crucially, he shows that model M2 does not use discrete latent variables when trained without labels. The reason this problem is not as severe in the GMVAE might possibly be the more restrictive assumptions in the generative process, which helps the optimisation, as argued in his blog.

Table 1: Unsupervised classification accuracy for MNIST with different numbers of clusters (K) (reported as percentage of correct labels)

Method	**K**	**Best Run**	**Average Run**
CatGAN (Springenberg, 2015)	20	90.30	-
AAE (Makhzani et al., 2015)	16	-	90.45 ± 2.05
AAE (Makhzani et al., 2015)	30	-	95.90 ± 1.13
DEC (Xie et al., 2015)	10	84.30	-
GMVAE (M = 1)	10	87.31	77.78 ± 5.75
GMVAE (M = 10)	10	88.54	82.31 ± 3.75
GMVAE (M = 1)	16	89.01	85.09 ± 1.99
GMVAE (M = 10)	16	96.92	87.82 ± 5.33
GMVAE (M = 1)	30	95.84	92.77 ± 1.60
GMVAE (M = 10)	30	93.22	89.27 ± 2.50

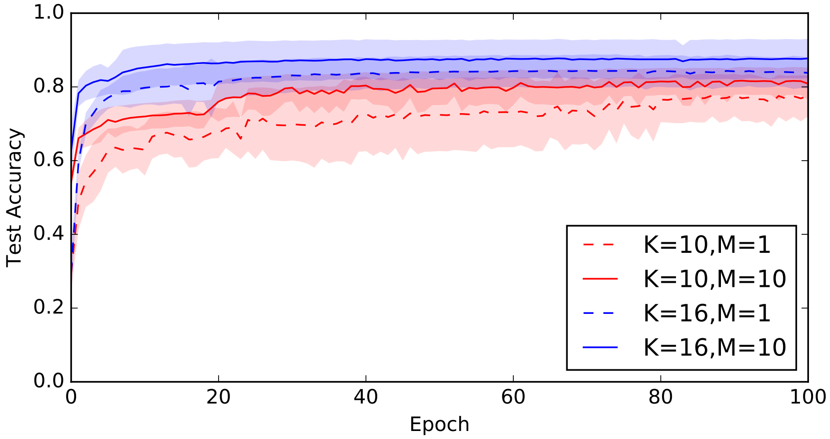


Figure 4: **Clustering Accuracy with different numbers of clusters (K) and Monte Carlo samples (M)** : After only few epochs, the GMVAE converges to a solution. Increasing the number of clusters improves the quality of the solution considerably.

4.2.1 Image Generation

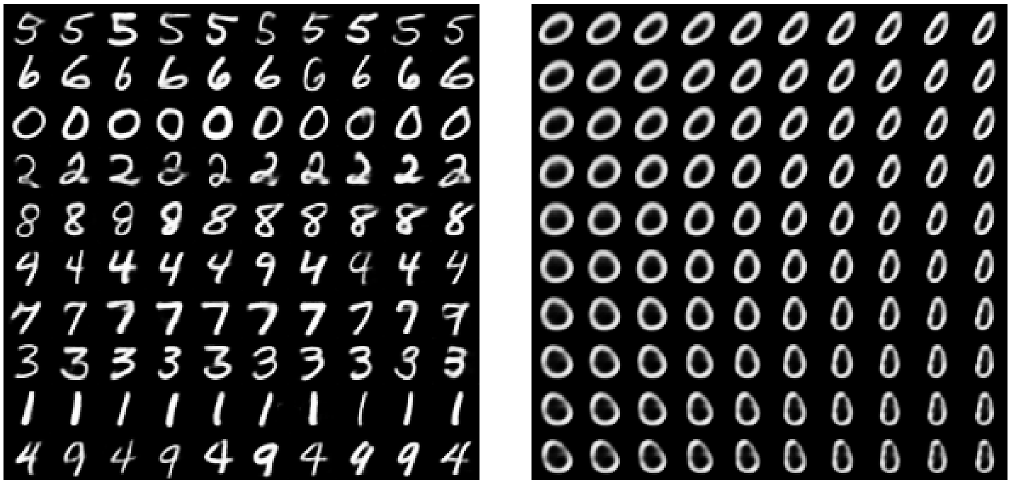
So far we have argued that the GMVAE picks up natural clusters in the dataset, and that these clusters share some structure with the actual classes of the images. Now we train the GMVAE with $K = 10$ on MNIST to show that the learnt components in the distribution of the latent space actually represent meaningful properties of the data. First, we note that there are two sources of stochasticity in play when sampling from the GMVAE, namely

1. Sampling $\boldsymbol{w}$ from its prior, which will generate the means and variances of $\boldsymbol{x}$ through a neural network β; and
2. Sampling $\boldsymbol{x}$ from the Gaussian mixture determined by $\boldsymbol{w}$ and $\boldsymbol{z}$, which will generate the image through a neural network θ.

In Fig. 5a we explore the latter option by setting $\boldsymbol{w} = 0$ and sampling multiple times from the resulting Gaussian mixture. Each row in Fig. 5a corresponds to samples from a different component of the Gaussian mixture, and it can be clearly seen that samples from the same component consistently result in images from the same class of digit. This confirms that the learned latent representation contains well differentiated clusters, and exactly one per digit. Additionally, in Fig. 5b we explore the sensitivity of the generated image to the Gaussian mixture components by smoothly varying

$\boldsymbol{w}$ and sampling from the same component. We see that while $\boldsymbol{z}$ reliably controls the class of the generated image, $\boldsymbol{w}$ sets the "style" of the digit.

Finally, in Fig. 6 we show images sampled from a GMVAE trained on SVHN, showing that the GMVAE clusters visually similar images together.



(a) Varying z

(b) Varying w

Figure 5: **Generated MNIST samples**: (a) Each row contains 10 randomly generated samples from different Gaussian components of the Gaussian mixture. The GMVAE learns a meaningful generative model where the discrete latent variables z correspond directly to the digit values in an unsupervised manner. (b) Samples generated by traversing around w space, each position of w correspond to a specific style of the digit.

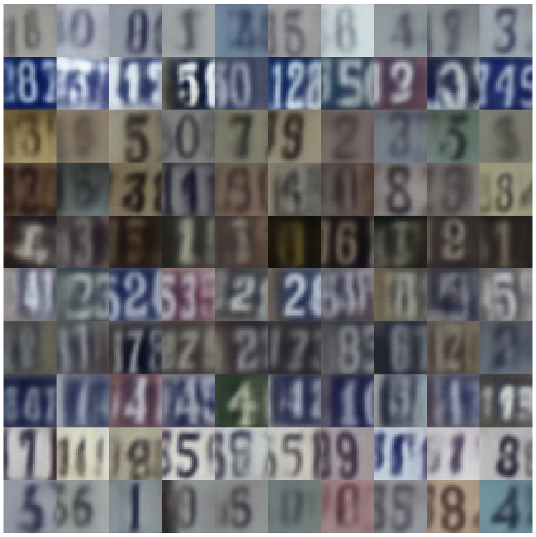


Figure 6: **Generated SVHN samples**: Each row corresponds to 10 samples generated randomly from different Gaussian components. GMVAE groups together images that are visually similar.

5 Conclusion

We have introduced a class of variational autoencoders in which one level of the latent encoding space has the form of a Gaussian mixture model, and specified a generative process that allows us to formulate a variational Bayes optimisation objective. We then discuss the problem of over-regularisation in VAEs. In the context of our model, we show that this problem manifests itself in the form of cluster degeneracy. Crucially, we show that this specific manifestation of the problem can be solved with standard heuristics.

We evaluate our model on unsupervised clustering tasks using popular datasets and achieving competitive results compared to the current state of the art. Finally, we show via sampling from the generative model that the learned clusters in the latent representation correspond to meaningful features of the visible data. Images generated from the same cluster in latent space share relevant high-level features (e.g. correspond to the same MNIST digit) while being trained in an entirely unsupervised manner.

It is worth noting that GMVAEs can be stacked by allowing the prior on w to be a Gaussian mixture distribution as well. A deep GMVAE could scale much better with number of clusters given that it would be combinatorial with regards to both number of layers and number of clusters per layer. As such, while future research on deep GMVAEs for hierarchical clustering is a possibility, it is crucial to also address the enduring optimisation challenges associated with VAEs in order to do so.

ACKNOWLEDGMENTS

We would like to acknowledge the NVIDIA Corporation for the donation of a GeForce GTX Titan Z used in our experiments. We would like to thank Jason Rolfe, Rui Shu and the reviewers for useful comments. Importantly, we would also like to acknowledge that the variational family which we used throughout this version of the paper was suggested by an anonymous reviewer.

REFERENCES

Charu C Aggarwal and Chandan K Reddy. *Data clustering: algorithms and applications*. CRC Press, 2013.

Christopher M Bishop. Pattern recognition and machine learning. 2006.

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. *arXiv preprint arXiv:1606.03657*, 2016a.

Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016b.

J. Chung, K. Kastner, L. Dinh, K. Goel, A. Courville, and Y. Bengio. A Recurrent Latent Variable Model for Sequential Data. *ArXiv e-prints*, June 2015.

SM Eslami, Nicolas Heess, Theophane Weber, Yuval Tassa, Koray Kavukcuoglu, and Geoffrey E Hinton. Attend, infer, repeat: Fast scene understanding with generative models. *arXiv preprint arXiv:1603.08575*, 2016.

PW Glynn. Likelihood ratio gradient estimation for stochastic systems. *Communications of the ACM*, 33(10):75–84, 1990.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2672–2680, 2014.

Alex Graves. Stochastic backpropagation through mixture density distributions. *arXiv preprint arXiv:1607.05690*, 2016.

Klaus Greff, Antti Rasmus, Mathias Berglund, Tele Hotloo Hao, Jürgen Schmidhuber, and Harri Valpola. Tagger: Deep unsupervised perceptual grouping. *arXiv preprint arXiv:1606.06724*, 2016.

Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. In *Proceedings of The 32nd International Conference on Machine Learning*, pp. 1462–1471, 2015.

I. Higgins, L. Matthey, X. Glorot, A. Pal, B. Uria, C. Blundell, S. Mohamed, and A. Lerchner. Early Visual Concept Learning with Unsupervised Deep Learning. *ArXiv e-prints*, June 2016.

Matthew D. Hoffman and Matthew J. Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. *Workshop in Advances in Approximate Bayesian Inference, NIPS*, 2016.

Matthew J Johnson, David Duvenaud, Alexander B Wiltschko, Sandeep R Datta, and Ryan P Adams. Composing graphical models with neural networks for structured representations and fast inference. *arXiv preprint arXiv:1603.06277*, 2016.

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

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

Diederik P Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems*, pp. 3581–3589, 2014.

Diederik P Kingma, Tim Salimans, and Max Welling. Improving variational inference with inverse autoregressive flow. *arXiv preprint arXiv:1606.04934*, 2016.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, and Ian Goodfellow. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*, 2014.

R. Shu, J. Brofos, F. Zhang, M. Ghavamzadeh, H. Bui, and M. Kochenderfer. Stochastic video prediction with conditional density estimation. In *European Conference on Computer Vision (ECCV) Workshop on Action and Anticipation for Visual Learning*, 2016.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. How to train deep variational autoencoders and probabilistic ladder networks. *arXiv preprint arXiv:1602.02282*, 2016.

Jost Tobias Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *arXiv preprint arXiv:1511.06390*, 2015.

Michalis Titsias and Miguel Lázaro-Gredilla. Local expectation gradients for black box variational inference. In *Advances in Neural Information Processing Systems*, pp. 2638–2646, 2015.

Martin J Wainwright and Michael I Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning*, 1(1-2):1–305, 2008.

Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. *arXiv preprint arXiv:1511.06335*, 2015.

A Network Parameters

For optimisation, we use Adam (Kingma & Ba, 2014) with a learning rate of 10^{-4} and standard hyperparameter values $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 10^{-8}$. The model architectures used in our experiments are shown in Tables A.1, A.2 and A.3.

Table A.1: **Neural network architecture models of $q_\phi(\boldsymbol{x}, \boldsymbol{w})$**: The hidden layers are shared between $q(\boldsymbol{x})$ and $q(\boldsymbol{w})$, except the output layer where the neural network is split into 4 output streams, 2 with dimension N_x and the other 2 with dimension N_w. We exponentiate the variance components to keep their value positive. An asterisk (*) indicates the use of batch normalization and a ReLU nonlinearity. For convolutional layers, the numbers in parentheses indicate stride-padding.

Dataset	Input	Hidden	Output
Synthetic	2	fc 120 ReLU 120 ReLU	N_w = 2, N_w = 2 (Exp), N_x = 2, N_x = 2 (Exp)
MNIST	28x28	conv 16x6x6* (1-0) 32x6x6* (1-0) 64x4x4* (2-1) 500*	N_w = 150, N_w = 150 (Exp), N_x = 200, N_x = 200 (Exp)
SVHN	32x32	conv 64x4x4* (2-1) 128x4x4* (2-1) 246x4x4* (2-1) 500*	N_w = 150, N_w = 150 (Exp), N_x = 200, N_x = 200 (Exp)

Table A.2: **Neural network architecture models of $p_\beta(\boldsymbol{x}|\boldsymbol{w}, \boldsymbol{z})$**: The output layers are split into $2K$ streams of output, where K streams return mean values and the other K streams output variances of all the clusters.

Dataset	Input	Hidden	Output
Synthetic	2	fc 120 Tanh	$\{N_x = 2\}_{2K}$
MNIST	150	fc 500 Tanh	$\{N_x = 200\}_{2K}$
SVHN	150	fc 500 Tanh	$\{N_x = 200\}_{2K}$

Table A.3: **Neural network architecture models of $p_\theta(\boldsymbol{y}|\boldsymbol{x})$**: The network outputs are Gaussian parameters for the synthetic dataset and Bernoulli parameters for MNIST and SVHN, where we use the logistic function to keep value of Bernoulli parameters between 0 and 1. An asterisk (*) indicates the use of batch normalization and a ReLU nonlinearity. For convolutional layers, the numbers in parentheses indicate stride-padding.

Dataset	Input	Hidden	Output
Synthetic	2	fc 120 ReLU 120 ReLU	$\{2\}_2$
MNIST	200	500* full-conv 64x4x4* (2-1) 32x6x6* (1-0) 16x6x6* (1-0)	28x28 (Sigmoid)
SVHN	200	500* full-conv 246x4x4* (2-1) 128x4x4* (2-1) 64x4x4* (2-1)	32x32 (Sigmoid)