Semi-Supervised Generation with Cluster-aware Generative Models

Maaløe, Lars; Fraccaro, Marco; Winther, Ole

Published in:
arXiv

Publication date:
2017

Document Version
Publisher's PDF, also known as Version of record

Semi-Supervised Generation with Cluster-aware Generative Models

Lars Maaløe [1] **Marco Fraccaro** [1] **Ole Winther** [1]

Abstract

Deep generative models trained with large amounts of unlabelled data have proven to be powerful within the domain of unsupervised learning. Many real life data sets contain a small amount of labelled data points, that are typically disregarded when training generative models. We propose the *Cluster-aware Generative Model*, that uses unlabelled information to infer a latent representation that models the natural clustering of the data, and additional labelled data points to refine this clustering. The generative performances of the model significantly improve when labelled information is exploited, obtaining a log-likelihood of -79.38 nats on permutation invariant MNIST, while also achieving competitive semi-supervised classification accuracies. The model can also be trained fully unsupervised, and still improve the log-likelihood performance with respect to related methods.

1. Introduction

Variational Auto-Encoders (VAE) (Kingma, 2013; Rezende et al., 2014) and Generative Adversarial Networks (GAN) (Goodfellow et al., 2014) have shown promising generative performances on data from complex high-dimensional distributions. Both approaches have spawn numerous related deep generative models, not only to model data points like those in a large unlabelled training data set, but also for semi-supervised classification (Kingma et al., 2014; Maaløe et al., 2016; Springenberg, 2015; Salimans et al., 2016). In semi-supervised classification a few points in the training data are endowed with class labels, and the plethora of unlabelled data aids to improve a supervised classification model.

Could a few labelled training data points in turn improve a deep generative model? This reverse perspective, doing semi-supervised generation, is investigated in this work. Many of the real life data sets contain a small amount of labelled data, but incorporating this partial knowledge in the generative models is not straightforward, because of the risk of overfitting towards the labelled data. This overfitting can be avoided by finding a good scheme for updating the parameters, like the one introduced in the models for semi-supervised classification (Kingma et al., 2014; Maaløe et al., 2016). However, there is a difference in optimizing the model towards optimal classification accuracy and generative performance. We introduce the *Cluster-aware Generative Model (CaGeM)*, an extension of a VAE, that improves the generative performances, by being able to model the natural clustering in the higher feature representations through a discrete variable (Bengio et al., 2013). The model can be trained fully unsupervised, but its performances can be further improved using labelled class information that helps in constructing well defined clusters. A generative model with added labelled data information may be seen as parallel to how humans rely on abstract domain knowledge in order to efficiently infer a causal model from property induction with very few labelled observations (Tenenbaum et al., 2006).

Supervised deep learning models with no stochastic units are able to learn multiple levels of feature abstraction. In VAEs, however, the addition of more stochastic layers is often accompanied with a built-in pruning effect so that the higher layers become disconnected and therefore not exploited by the model (Burda et al., 2015a; Sønderby et al., 2016). As we will see, in CaGeM the possibility of learning a representation in the higher stochastic layers that can model clusters in the data drastically reduces this issue. This results in a model that is able to disentangle some of the factors of variation in the data and that extracts a hierarchy of features beneficial during the generation phase. By using only 100 labelled data points, we present state of the art log-likelihood performance on permutation-invariant models for MNIST, and an improvement with respect to comparable models on the OMNIGLOT data set. While the main focus of this paper is semi-supervised generation, we also show that the same model is able to achieve competitive semi-supervised classification results.

[1]Technical University of Denmark. Correspondence to: Lars Maaløe <larsma@dtu.dk>, Marco Fraccaro <marfra@dtu.dk>, Ole Winther <olwi@dtu.dk>.

2. Variational Auto-encoders

A *Variational Auto-Encoder (VAE)* (Kingma, 2013; Rezende et al., 2014) defines a deep generative model for data x that depends on latent variable z or a hierarchy of latent variables, e.g. $z = [z_1, z_2]$, see Figure 1a for a graphical representation. The joint distribution of the two-level generative model is given by

$$p_\theta(x, z_1, z_2) = p_\theta(x|z_1)p_\theta(z_1|z_2)p(z_2) \ ,$$

where

$$\begin{aligned} p_\theta(z_1|z_2) &= \mathcal{N}(z_1; \mu^1_\theta(z_2), \sigma^1_\theta(z_2)) \\ p(z_2) &= \mathcal{N}(z_2; 0, I) \end{aligned}$$

are Gaussian distributions with a diagonal covariance matrix and $p_\theta(x|z_1)$ is typically a parameterized Gaussian (continuous data) or Bernoulli distribution (binary data). The probability distributions of the generative model of a VAE are parameterized using deep neural networks whose parameters are denoted by θ. Training is performed by optimizing the *Evidence Lower Bound (ELBO)*, a lower bound to the intractable log-likelihood $\log p_\theta(x)$ obtained using Jensen's inequality:

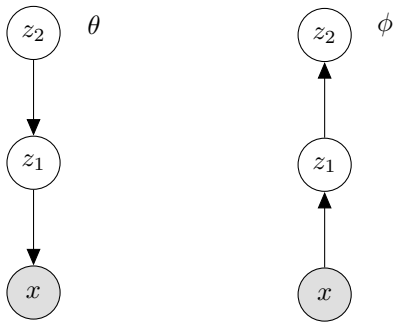
$$\begin{aligned} \log p_\theta(x) &= \log \iint p_\theta(x, z_1, z_2) \mathrm{d}z_1 \mathrm{d}z_2 \\ &\geq \mathbb{E}_{q_\phi(z_1, z_2|x)} \left[\log \frac{p_\theta(x, z_1, z_2)}{q_\phi(z_1, z_2|x)} \right] = \mathcal{F}(\theta, \phi) \ . \end{aligned} \tag{1}$$

The introduced variational distribution $q_\phi(z_1, z_2|x)$ is an approximation to the model's posterior distribution $p_\theta(z_1, z_2|x)$, defined with a bottom-up dependency structure where each variable of the model depends on the variable below in the hierarchy:

$$\begin{aligned} q_\phi(z_1, z_2|x) &= q_\phi(z_1|x)q_\phi(z_2|z_1) \\ q_\phi(z_1|x) &= \mathcal{N}(z_1; \mu^1_\phi(x), \sigma^1_\phi(x)) \\ q_\phi(z_2|z_1) &= \mathcal{N}(z_2; \mu^2_\phi(z_1), \sigma^2_\phi(z_1)) \ . \end{aligned}$$

Similar to the generative model, the mean and diagonal covariance of both Gaussian distributions defining the inference network q_ϕ are parameterized with deep neural networks that depend on parameters ϕ, see Figure 1b for a graphical representation.

We can learn the parameters θ and ϕ by jointly maximizing the ELBO $\mathcal{F}(\theta, \phi)$ in (1) with stochastic gradient ascent, using Monte Carlo integration to approximate the intractable expectations and computing low variance gradients with the reparameterization trick (Kingma, 2013; Rezende et al., 2014).



(a) Generative model p_θ (b) Inference model q_ϕ

Figure 1: Generative model and inference model of a Variational Auto-Encoder with two stochastic layers.

Inactive stochastic units A common problem encountered when training VAEs with bottom-up inference networks is given by the so called *inactive units* in the higher layers of stochastic variables (Burda et al., 2015a; Sønderby et al., 2016). In a 2-layer model for example, VAEs often learn $q_\phi(z_2|z_1) = p(z_2) = \mathcal{N}(z_2; 0, I)$, i.e. the variational approximation of z_2 uses no information coming from the data point x through z_1. If we rewrite the ELBO in (1) as

$$\begin{aligned} \mathcal{F}(\theta, \phi) = \mathbb{E}_{q_\phi(z_1, z_2|x)} \left[\log \frac{p_\theta(x, z_1|z_2)}{q_\phi(z_1|x)} \right] - \\ \mathbb{E}_{q_\phi(z_1|x)} \left[KL \left[q_\phi(z_2|z_1) || p(z_2) \right] \right] \end{aligned}$$

we can see that $q_\phi(z_2|z_1) = p(z_2)$ represents a local maxima of our optimization problem where the KL-divergence term is set to zero and the information flows by first sampling in $\widetilde{z_1} \sim q_\phi(z_1|x)$ and then computing $p_\theta(x|\widetilde{z_1})$ (and is therefore independent from z_2). Several techniques have been developed in the literature to mitigate the problem of inactive units, among which we find annealing of the KL term (Bowman et al., 2015; Sønderby et al., 2016) or the use of free bits (Kingma et al., 2016).

Using ideas from Chen et al. (2017), we notice that the inactive units in a VAE with 2 layers of stochastic units can be justified not only as a poor local maxima, but also from the modelling point of view. Chen et al. (2017) give a *bits-back coding* interpretation of Variational Inference for a generative model of the form $p(x, z) = p(x|z)p(z)$, with data x and stochastic units z. The paper shows that if the decoder $p(x|z)$ is powerful enough to explain most of the structure in the data (e.g. an autoregressive decoder), then it will be convenient for the model to set $q(z|x) = p(z)$ not to incur in an extra optimization cost of $KL[q(z|x)||p(z|x)]$. The inactive z_2 units in a 2-layer VAE can therefore be seen as caused by the flexible distribution $p_\theta(x, z_1|z_2)$ that is able to explain most of the structure in the data without using information from z_2. By making $q_\phi(z_2|z_1) = p(z_2)$, the

model can avoid the extra cost of $KL\left[q_\phi(z_2|x)||p_\theta(z_2|x)\right]$. A more detailed discussion on the topic can be found in Appendix A.

It is now clear that if we want a VAE to exploit the power of additional stochastic layers we need to define it so that the benefits of encoding meaningful information in z_2 is greater than the cost $KL\left[q_\phi(z_2|x)||p_\theta(z_2|x)\right]$ that the model has to pay. As we will discuss below, we will achieve this by aiding the generative model to do representation learning.

3. Cluster-aware Generative Models

Hierarchical models parameterized by deep neural networks have the ability to represent very flexible distributions. However, in the previous section we have seen that the units in the higher stochastic layers of a VAE often become inactive. We will show that we can help the model to exploit the higher stochastic layers by explicitly encoding a useful representation, i.e. the ability to model the natural clustering of the data (Bengio et al., 2013), which will also be needed for semi-supervised generation.

We favor the flow of higher-level global information through z_2 by extending the generative model of a VAE with a discrete variable y representing the choice of one out of K different clusters in the data. The joint distribution $p_\theta(x, z_1, z_2)$ is computed by marginalizing over y:

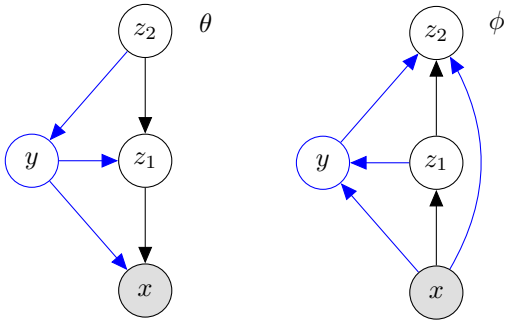
$$\begin{aligned} p_\theta(x, z_1, z_2) &= \sum_y p_\theta(x, y, z_1, z_2) \\ &= \sum_y p_\theta(x|y, z_1) p_\theta(z_1|y, z_2) p_\theta(y|z_2) p(z_2) \ . \end{aligned}$$

We call this model *Cluster-aware Generative Model (CaGeM)*, see Figure 2 for a graphical representation. The introduced categorical distribution $p_\theta(y|z_2) = \text{Cat}(y; \pi_\theta(z_2))$ (π_θ represents the class distribution) depends solely on z_2, that needs therefore to stay active for the model to be able to represent clusters in the data. We further add the dependence of z_1 and x on y, so that they can now both also represent cluster-dependent information.

3.1. Inference

As done for the VAE in (1), we can derive the ELBO for CaGeM by maximizing the log-likelihood

$$\begin{aligned} \log p_\theta(x) &= \log \iint p_\theta(x, z_1, z_2) \mathrm{d}z_1 \mathrm{d}z_2 \\ &= \log \iint \sum_y p_\theta(x, y, z_1, z_2) \mathrm{d}z_1 \mathrm{d}z_2 \\ &\geq \mathbb{E}_{q_\phi(y, z_1, z_2|x)} \left[\log \frac{p_\theta(x, y, z_1, z_2)}{q_\phi(y, z_1, z_2|x)} \right] . \end{aligned}$$



(a) Generative model p_θ (b) Inference model q_ϕ

Figure 2: Generative model and inference model of a CaGeM with two stochastic layers (black and blue lines). The black lines only represent a standard VAE.

We define the variational approximation $q_\phi(y, z_1, z_2|x)$ over the latent variables of the model as

$$q_\phi(y, z_1, z_2|x) = q_\phi(z_2|x, y, z_1) q_\phi(y|z_1, x) q_\phi(z_1|x) \ ,$$

where

$$\begin{aligned} q_\phi(z_1|x) &= \mathcal{N}(z_1; \mu^1_\phi(x), \sigma^1_\phi(x)) \\ q_\phi(z_2|x, y, z_1) &= \mathcal{N}(z_2; \mu^2_\phi(y, z_1), \sigma^2_\phi(y, z_1)) \\ q_\phi(y|z_1, x) &= \text{Cat}(y; \pi_\phi(z_1, x)) \end{aligned}$$

In the inference network we then reverse all the dependencies among random variables in the generative model (the arrows in the graphical model in Figure 2). This results in a *bottom-up* inference network that performs a feature extraction that is fundamental for learning a good representation of the data. Starting from the data x we construct higher levels of abstraction, first through the variables z_1 and y, and finally through the variable z_2, that includes the global information used in the generative model. In order to make the higher representation more expressive we add a skip-connection from x to z_2, that is however not fundamental to improve the performances of the model.

With this factorization of the variational distribution $q_\phi(y, z_1, z_2|x)$, the ELBO can be written as

$$\begin{aligned} \mathcal{F}(\theta, \phi) = \mathbb{E}_{q_\phi(z_1|x)} \Bigg[\sum_y q_\phi(y|z_1, x) \cdot \\ \cdot \mathbb{E}_{q_\phi(z_2|x, y, z_1)} \left[\log \frac{p_\theta(x, y, z_1, z_2)}{q_\phi(y, z_1, z_2|x)} \right] \Bigg] \ . \end{aligned}$$

We maximize $\mathcal{F}(\theta, \phi)$ by jointly updating, with stochastic gradient ascent, the parameters θ of the generative model and ϕ of the variational approximation. When computing the gradients, the summation over y is performed analytically, whereas the intractable expectations over z_1 and z_2

are approximated by sampling. We use the reparameterization trick to reduce the variance of the stochastic gradients.

4. Semi-Supervised Generation with CaGeM

In some applications we may have class label information for some of the data points in the training set. In the following we will show that CaGeM provides a natural way to exploit additional labelled data to improve the performance of the generative model. Notice that this *semi-supervised generation* approach differs from the more traditional *semi-supervised classification* task that uses unlabelled data to improve classification accuracies (Kingma et al., 2014; Maaløe et al., 2016; Salimans et al., 2016). In our case in fact, it is the labelled data that supports the generative task. Nevertheless, we will see in our experiment that CaGeM also leads to competitive semi-supervised classification performances.

To exploit the class information, we first set the number of clusters K equal to the number of classes C. We can now define two classifiers in CaGeM:

1. In the inference network we can compute the class probabilities given the data, i.e. $q_\phi(y|x)$, by integrating out the stochastic variables z_1 from $q_\phi(y, z_1|x)$

$$\begin{aligned} q_\phi(y|x) &= \int q_\phi(y, z_1|x)\mathrm{d}z_1 \\ &= \int q_\phi(y|z_1, x)q_\phi(z_1|x)\mathrm{d}z_1 \end{aligned}$$

2. Another set of class-probabilities can be computed using the generative model. Given the posterior distribution $p_\theta(z_2|x)$ we have in fact

$$p_\theta(y|x) = \int p_\theta(y|z_2)p_\theta(z_2|x)\mathrm{d}z_2 \ .$$

The posterior over z_2 is intractable, but we can approximate it using the variational approximation $q_\phi(z_2|x)$, that is obtained by marginalizing out y and the variable z_1 in the joint distribution $q_\phi(y, z_1, z_2|x)$:

$$\begin{aligned} p_\theta(y|x) &\approx \int p_\theta(y|z_2)q_\phi(z_2|x)\mathrm{d}z_2 \\ &= \int p_\theta(y|z_2)\left(\int \sum_{\widetilde{y}} q_\phi(z_2|x, \widetilde{y}, z_1)\cdot \right. \\ &\qquad \left. \cdot\, q_\phi(\widetilde{y}|z_1, x)q_\phi(z_1|x)\mathrm{d}z_1\right)\mathrm{d}z_2 \ . \end{aligned}$$

While for the labels $\widetilde{y}$ the summation can be carried out analytically, for the variable z_1 and z_2 we use Monte Carlo integration. For each of the C classes we will then obtain a different z_2^c sample $(c = 1, \dots C)$ with a corresponding weight given by $q_\phi(\widetilde{y}^c|z_1, x)$. This therefore resembles a *cascade* of classifiers, as the class probabilities of the $p_\theta(y|x)$ classifier will depend on the probabilities of the classifier $q_\phi(y|z_1, x)$ in the inference model.

As our main goal is to learn representations that will lead to good generative performance, we interpret the classification of the additional labelled data as a secondary task that aids in learning a z_2 feature space that can be easily separated into clusters. We can then see this as a form of *semi-supervised clustering* (Basu et al., 2002), where we know that some data points belong to the same cluster and we are free to learn a data manifold that makes this possible.

The optimal features for the classification task could be very different from the representations learned for the generative task. This is why it is important not to update the parameters of the distributions over z_1, z_2 and x, in both generative model and inference model, using labelled data information. If this is not done carefully, the model could be prone to overfitting towards the labelled data. We define as θ_y the subset of θ containing the parameters in $p_\theta(y|z_2)$, and as ϕ_y the subset of ϕ containing the parameters in $q_\phi(y|z_1, x)$. θ_y and ϕ_y then represent the incoming arrows to y in Figure 2. We update the parameters θ and ϕ jointly by maximizing the new objective

$$\mathcal{I} = \sum_{\{x_u\}} \mathcal{F}(\theta, \phi) - \alpha \left(\sum_{\{x_l, y_l\}} (\mathcal{H}_p(\theta_y, \phi_y) + \mathcal{H}_q(\phi_y)) \right)$$

where $\{x_u\}$ is the set of unlabelled training points, $\{x_l, y_l\}$ is the set of labelled ones, and $\mathcal{H}_p$ and $\mathcal{H}_q$ are the standard categorical cross-entropies for the $p_\theta(y|x)$ and $q_\phi(y|x)$ classifiers respectively. Notice that we consider the cross-entropies only a function of θ_y and ϕ_y, meaning that the gradients of the cross-entropies with respect to the parameters of the distributions over z_1, z_2 and x will be 0, and will not depend on the labelled data (as needed when learning meaningful representations of the data to be used for the generative task). To match the relative magnitudes between the ELBO $\mathcal{F}(\theta, \phi)$ and the two cross-entropies we set $\alpha = \beta \frac{N_u + N_l}{N_l}$ as done in (Kingma et al., 2014; Maaløe et al., 2016), where N_u and N_l are the numbers of unlabelled and labelled data points, and β is a scaling constant.

5. Experiments

We evaluate CaGeM by computing the generative log-likelihood performance on MNIST and OMNIGLOT (Lake et al., 2013) datasets. The model is parameterized by feed-forward neural networks (`NN`) and linear layers (`Linear`),

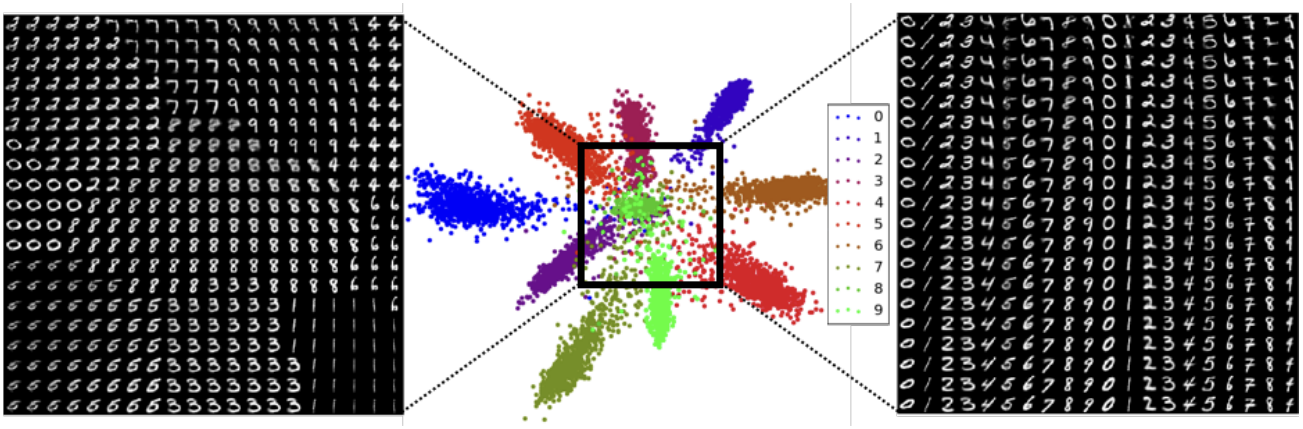


Figure 3: Visualizations from CaGeM-100 with a 2-dimensional z_2 space. The middle plot shows the latent space, from which we generate random samples (left) and class conditional random samples (right) with a mesh grid (black bounding box). The relative placement of the samples in the scatter plot corresponds to a digit in the mesh grid.

so that, for Gaussian outputs, each collection of incoming edges to a node in Figure 2 is defined as:

$$d = \texttt{NN}(x) \qquad \mu = \texttt{Linear}(d) \qquad \log \sigma = \texttt{Linear}(d)\ .$$

For Bernoulli distributed outputs we simply define a feed-forward neural network with a sigmoid activation function for the output. Between dense layers we use the rectified linear unit as non-linearity and batch-normalization (Ioffe & Szegedy, 2015). We only collect statistics for the batch-normalization during unlabelled inference. For the log-likelihood experiments we apply temperature on the KL-terms during the first 100 epochs of training (Bowman et al., 2015; Sønderby et al., 2016). The stochastic layers are defined with $\texttt{dim}(z_1) = 64$, $\texttt{dim}(z_2) = 32$ and 2-layered neural feed-forward networks with respectively 1024 and 512 units in each layer. Training is performed using the Adam optimizer (Kingma & Ba, 2014) with an initial learning rate of 0.001 and annealing it by .75 every 50 epochs. The experiments are implemented with Theano (Bastien et al., 2012), Lasagne (Dieleman et al., 2015) and Parmesan[1].

For both datasets we report unsupervised and semi-supervised permutation invariant log-likelihood performance and for MNIST we also report semi-supervised classification errors. The input data is dynamically binarized and the ELBO is evaluated by taking 5000 importance-weighted (IW) samples, denoted $\mathcal{F}_{5000}$. We evaluate the performance of CaGeM with different numbers of labelled samples referred to as CaGeM-#`labels`. When used, the labelled data is randomly sampled evenly across the class distribution. All experiments across datasets are run with the same architecture.

[1]A variational repository named parmesan on Github.

6. Results

Table 1 shows the generative log-likelihood performances of different variants of CaGeM on the MNIST data set. We can see that the more labelled samples we use, the better the generative performance will be. Even though the results are not directly comparable, since CaGeM exploits a small fraction supervised information, we find that using only 100 labelled samples (10 samples per class), CaGeM-100 model achieves state of the art log-likelihood performance on permutation invariant MNIST with a simple 2-layered model. We also trained a ADGM-100 from Maaløe et al. (2016)[2] in order to make a fair comparison on generative log-likelihood in a semi-supervised setting and reached a performance of -86.06 nats. This indicates that models that are highly optimized for improving semi-supervised classification accuracy may be a suboptimal choice for generative modeling.

CaGeM could further benefit from the usage of non-permutation invariant architectures suited for image data, such as the autoregressive decoders used by IAF VAE (Kingma et al., 2016) and VLAE (Chen et al., 2017). The fully unsupervised CaGeM-0 results show that by defining clusters in the higher stochastic units, we achieve better performances than the closely related IWAE (Burda et al., 2015a) and LVAE (Sønderby et al., 2016) models. It is finally interesting to see from Table 1 that CaGeM-0 performs well even when the number of clusters are different from the number of classes in the labelled data set.

In Figure 4 we show in detail how the performance of CaGeM increases as we add more labelled data points. We can also see that the ELBO $\mathcal{F}_1^{test}$ tightens when

[2]We used the code supplied in the repository named auxiliary-deep-generative-models on Github.

	$\leq \log p(x)$
NON-PERMUTATION INVARIANT	
DRAW+VGP (TRAN ET AL., 2016)	-79.88
IAF VAE (KINGMA ET AL., 2016)	-79.10
VLAE (CHEN ET AL., 2017)	-78.53
PERMUTATION INVARIANT	
AVAE, L=2, IW=1 (MAALØE ET AL., 2016)	-82.97
IWAE, L=2, IW=50 (BURDA ET AL., 2015A)	-82.90
LVAE, L=5, IW=10 (SØNDERBY ET AL., 2016)	-81.74
VAE+VGP, L=2 (TRAN ET AL., 2016)	-81.32
DVAE (ROLFE, 2017)	-80.04
CAGEM-0, L=2, IW=1, K=20	-82.18
CAGEM-0, L=2, IW=1, K=10	-81.60
CAGEM-20, L=2, IW=1	-81.47
CAGEM-50, L=2, IW=1	-80.49
CAGEM-100, L=2, IW=1	-79.38

Table 1: Test log-likelihood for permutation invariant and non-permutation invariant MNIST. L, IW and K denotes the number of stochastic layers (if it is translatable to the VAE), the number of importance weighted samples used during inference, and the number of predefined clusters used.

adding more labelled information, as compared to $\mathcal{F}_1^{\text{LVAE}} = -85.23$ and $\mathcal{F}_1^{\text{VAE}} = -87.49$ (Sønderby et al., 2016).

The PCA plots of the z_2 variable of a VAE, CaGeM-0 and CaGeM-100 are shown in Figure 5 . We see how CaGeMs encode clustered information into the higher stochastic layer. Since CaGeM-0 is unsupervised, it forms less class-dependent clusters compared to the semi-supervised CaGeM-100, that fits its z_2 latent space into 10 nicely separated clusters. Regardless of the labelled information added during inference, CaGeM manages to activate a high amount of units, as for CaGeM we obtain $KL[q_\phi(z_2|x,y)||p(z_2)] \approx 17$ nats, while a LVAE with 2 stochastic layers obtains ≈ 9 nats.

The generative model in CaGeM enables both random samples, by sampling the class variable $y \sim p_\theta(y|z_2)$ and feeding it to $p_\theta(x|z_1, y)$, and class conditional samples by fixing y. Figure 3 shows the generation of MNIST digits from CaGeM-100 with $\texttt{dim}(z_2) = 2$. The images are generated by applying a linearly spaced mesh grid within the latent space z_2 and performing random generations (left) and conditional generations (right). When generating samples in CaGeM, it is clear how the latent units z_1 and z_2 capture different modalities within the true data distribution, namely style and class.

Regardless of the fact that CaGeM was designed to optimize the semi-supervised generation task, the model can also be used for classification by using the classifier $p_\theta(y|x)$. In Table 2 we show that the semi-supervised classification accuracies obtained with CaGeM are comparable

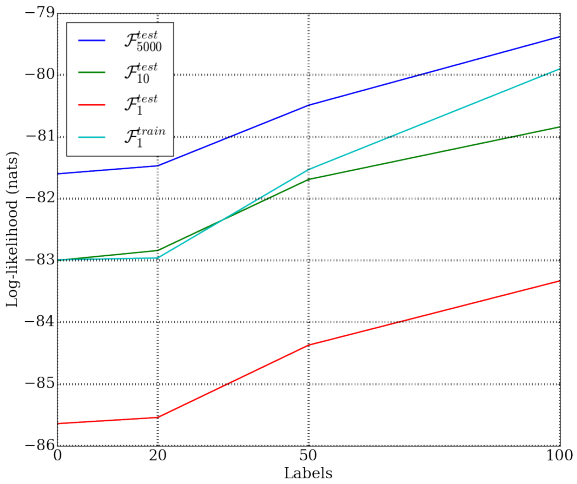


Figure 4: Log-likelihood scores for CaGeM on MNIST with 0, 20, 50 and 100 labels with 1, 10 and 5000 IW samples.

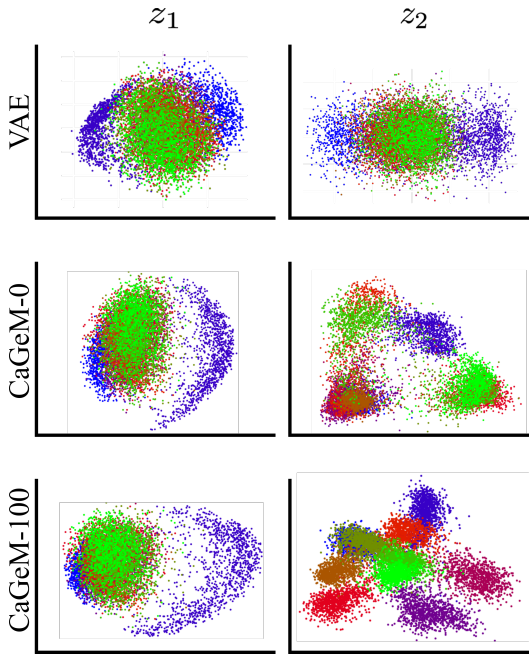


Figure 5: PCA plots of the stochastic units z_1 and z_2 in a 2-layered model trained on MNIST. The colors corresponds to the true labels.

to the performance of GANs (Salimans et al., 2016).

The OMNIGLOT dataset consists of 50 different alphabets of handwritten characters, where each character is sparsely represented. In this task we use the alphabets as the cluster information, so that the z_2 representation should divide correspondingly. From Table 3 we see an improvement

LABELS	20	50	100
M1+M2 (KINGMA ET AL., 2014)	-	-	3.33% (±0.14)
VAT (MIYATO ET AL., 2015)	-	-	2.12%
CATGAN (SPRINGENBERG, 2015)	-	-	1.91% (±0.1)
SDGM (MAALØE ET AL., 2016)	-	-	1.32% (±0.07)
LADDER NETWORK (RASMUS ET AL., 2015)	-	-	1.06% (±0.37)
ADGM (MAALØE ET AL., 2016)	-	-	0.96% (±0.02)
IMP. GAN (SALIMANS ET AL., 2016)	16.77% (±4.52)	2.21% (±1.36)	0.93% (±0.65)
CAGEM	15.86%	2.42%	1.16%

Table 2: Semi-supervised test error % benchmarks on MNIST for 20, 50, and 100 randomly chosen and evenly distributed labelled samples. Each experiment was run 3 times with different labelled subsets and the reported accuracy is the mean value.

over other comparable VAE architectures (VAE, IWAE and LVAE), however, the performance is far from the once reported from the auto-regressive models (Kingma et al., 2016; Chen et al., 2017). This indicates that the alphabet information is not as strong as for a dataset like MNIST. This is also indicated from the accuracy of CaGeM-500, reaching a performance of $\approx 24\%$. Samples from the model can be found in Figure 6.

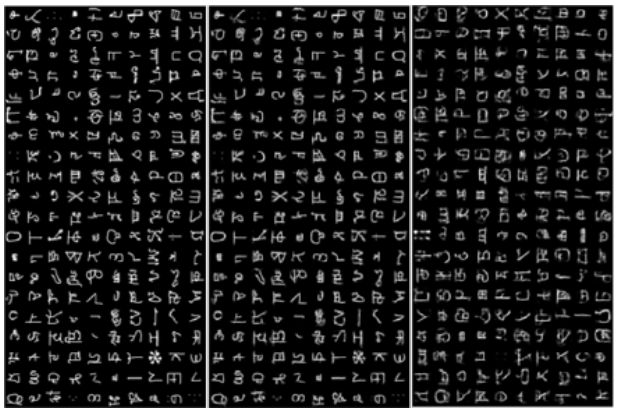


Figure 6: Generations from CaGeM-500. (left) The input images, (middle) the reconstructions, and (right) random samples from z_2.

	$\leq \log p(x)$
VAE, L=2, IW=50 (BURDA ET AL., 2015A)	−106.30
IWAE, L=2, IW=50 (BURDA ET AL., 2015A)	−103.38
LVAE, L=5, FT, IW=10 (SØNDERBY ET AL., 2016)	−102.11
RBM (BURDA ET AL., 2015B)	−100.46
DBN (BURDA ET AL., 2015B)	−100.45
DVAE (ROLFE, 2017)	−97.43
CAGEM-500, L=2, IW=1	−100.86

Table 3: Generative test log-likelihood for permutation invariant OMNIGLOT.

7. Discussion

As we have seen from our experiments, CaGeM offers a way to exploit the added flexibility of a second layer of stochastic units, that stays active as the modeling performances can greatly benefit from capturing the natural clustering of the data. Other recent works have presented alternative methods to mitigate the problem of inactive units when training flexible models defined by a hierarchy of stochastic layers. Burda et al. (2015a) used importance samples to improve the tightness of the ELBO, and showed that this new training objective helped in activating the units of a 2-layer VAE. Sønderby et al. (2016) trained Ladder Variational Autoencoders (LVAE) composed of up to 5 layers of stochastic units, using a top-down inference network that forces the information to flow in the higher stochastic layers. Contrarily to the bottom-up inference network of CaGeM, the top-down approach used in LVAEs does not enforce a clear separation between the role of each stochastic unit, as proven by the fact that all of them encode some class information. Longer hierarchies of stochastic units unrolled in time can be found in the sequential setting (Krishnan et al., 2015; Fraccaro et al., 2016). In these applications the problem of inactive stochastic units appears when using powerful autoregressive decoders (Fraccaro et al., 2016; Chen et al., 2017), but is mitigated by the fact that new data information enters the model at each time step.

The discrete variable y of CaGeM was introduced to be able to define a better learnable representation of the data, that helps in activating the higher stochastic layer. The combination of discrete and continuous variables for deep generative models was also recently explored by several authors. Maddison et al. (2016); Jang et al. (2016) used a continuous relaxation of the discrete variables, that makes it possible to efficiently train the model using stochastic backpropagation. The introduced Gumbel-Softmax variables allow to sacrifice log-likelihood performances to avoid the computationally expensive integration over y. Rolfe (2017) presents a new class of probabilistic models that combines an undirected component consisting of a bipartite Boltzmann machine with binary units and a directed component with multiple layers of continuous variables.

Traditionally, semi-supervised learning applications of deep generative models such as Variational Auto-encoders and Generative Adversarial Networks (Goodfellow et al., 2014) have shown that, whenever only a small fraction of labelled data is available, the supervised classification task can benefit from additional unlabelled data (Kingma et al., 2014; Maaløe et al., 2016; Salimans et al., 2016). In this work we consider the semi-supervised problem from a different perspective, and show that the generative task of CaGeM can benefit from additional labelled data. As a by-product of our model however, we also obtain competitive semi-supervised classification results, meaning that CaGeM is able to share statistical strength between the generative and classification tasks.

When modeling natural images, the performance of CaGeM could be further improved using more powerful autoregressive decoders such as the ones in (Gulrajani et al., 2016; Chen et al., 2017). Also, an even more flexible variational approximation could be obtained using auxiliary variables (Ranganath et al., 2015; Maaløe et al., 2016) or normalizing flows (Rezende & Mohamed, 2015; Kingma et al., 2016).

8. Conclusion

In this work we have shown how to perform semi-supervised generation with CaGeM. We showed that CaGeM improves the generative log-likelihood performance over similar deep generative approaches by creating clusters for the data in its higher latent representations using unlabelled information. CaGeM also provides a natural way to refine the clusters using additional labelled information to further improve its modelling power.

A. The Problem of Inactive Units

First consider a model $p(x)$ without latent units. We consider the asymptotic average properties, so we take the expectation of the log-likelihood over the (unknown) data distribution $p_{\text{data}}(x)$:

$$\begin{aligned}\mathbb{E}_{p_{\text{data}}(x)}\left[\log p(x)\right] &= \mathbb{E}_{p_{\text{data}}(x)}\left[\log\left(p_{\text{data}}(x)\frac{p(x)}{p_{\text{data}}(x)}\right)\right] \\ &= -\mathcal{H}(p_{\text{data}}) - KL(p_{\text{data}}(x)||p(x)) \,,\end{aligned}$$

where $\mathcal{H}(p) = -\mathbb{E}_{p(x)}\left[\log p(x)\right]$ is the entropy of the distribution and $KL(\cdot||\cdot)$ is the KL-divergence. The expected log-likelihood is then simply the baseline entropy of the data generating distribution minus the deviation between the data generating distribution and our model for the distribution.

For the latent variable model $p_{\text{lat}}(x) = \int p(x|z)p(z)dz$ the log-likelihood bound is:

$$\log p_{\text{lat}}(x) \geq \mathbb{E}_{q(z|x)}\left[\log\frac{p(x|z)p(z)}{q(z|x)}\right] \,.$$

We take the expectation over the data generating distribution and apply the same steps as above

$$\begin{aligned}\mathbb{E}_{p_{\text{data}}(x)}\left[\log p_{\text{lat}}(x)\right] &\geq \mathbb{E}_{p_{\text{data}}(x)}\mathbb{E}_{q(z|x)}\left[\log\frac{p(x|z)p(z)}{q(z|x)}\right] \\ &= -\mathcal{H}(p_{\text{data}}) - KL(p_{\text{data}}(x)||p_{\text{lat}}(x)) \\ &\quad - \mathbb{E}_{p_{\text{data}}(x)}\left[KL(q(z|x)||p(z|x))\right] \,,\end{aligned}$$

where $p(z|x) = p(x|z)p(z)/p_{\text{lat}}(x)$ is the (intractable) posterior of the latent variable model. This results shows that we pay an additional price (the last term) for using an approximation to the posterior.

The latent variable model can choose to ignore the latent variables, $p(x|z) = \hat{p}(x)$. When this happens the expression falls back to the log-likelihood without latent variables. We can therefore get an (intractable) condition for when it is advantageous for the model to use the latent variables:

$$\begin{aligned}\mathbb{E}_{p_{\text{data}}(x)}\left[\log p_{\text{lat}}(x))\right] > \mathbb{E}_{p_{\text{data}}(x)}\left[\log \hat{p}(x))\right] + \\ \mathbb{E}_{p_{\text{data}}(x)}\left[KL(q(z|x)||p(z|x))\right] \,.\end{aligned}$$

The model will use latent variables when the log-likelihood gain is so high that it can compensate for the loss $KL(q(z|x)||p_{\text{lat}}(z|x))$ we pay by using an approximate posterior distribution.

The above argument can also be used to understand why it is harder to get additional layers of latent variables to become active. For a two-layer latent variable model $p(x, z_1, z_2) = p(x|z_1)p(z_1|z_2)p(z_2)$ we use a variational distribution $q(z_1, z_2|x) = q(z_2|z_1, x)q(z_1|x)$ and decompose the log likelihood bound using $p(x, z_1, z_2) = p(z_2|z_1, x)p(z_1|x)p_{\text{lat},2}(x)$:

$$\begin{aligned}&\mathbb{E}_{p_{\text{data}}(x)}\left[\log p_{\text{lat},2}(x)\right] \\ &\quad\geq \mathbb{E}_{p_{\text{data}}(x)}\mathbb{E}_{q(z_1,z_2|x)}\left[\log\frac{p(x|z_1)p(z_1|z_2)p(z_2)}{q(z_1,z_2|x)}\right] \\ &\quad= -H(p_{\text{data}}) - KL(p_{\text{data}}(x)||p_{\text{lat},2}(x)) \\ &\qquad- \mathbb{E}_{p_{\text{data}}(x)}\mathbb{E}_{q(z_1|x)}\left[KL(q(z_2|z_1,x)||p(z_2|z_1,x))\right] \\ &\qquad- \mathbb{E}_{p_{\text{data}}(x)}KL(q(z_1|x)||p(z_1|x)) \,.\end{aligned}$$

Again this expression falls back to the one-layer model when $p(z_1|z_2) = \hat{p}(z_1)$. So whether to use the second layer of stochastic units will depend upon the potential diminishing return in terms of log likelihood relative to the extra KL-cost from the approximate posterior.

Acknowledgements

We thank Ulrich Paquet for fruitful feedback. The research was supported by Danish Innovation Foundation, the NVIDIA Corporation with the donation of TITAN X GPUs. Marco Fraccaro is supported by Microsoft Research through its PhD Scholarship Programme.

References

Bastien, Frédéric, Lamblin, Pascal, Pascanu, Razvan, Bergstra, James, Goodfellow, Ian J., Bergeron, Arnaud, Bouchard, Nicolas, and Bengio, Yoshua. Theano: new features and speed improvements. In *Deep Learning and Unsupervised Feature Learning, workshop at Neural Information Processing Systems*, 2012.

Basu, Sugato, Banerjee, Arindam, and Mooney, Raymond J. Semi-supervised clustering by seeding. In *Proceedings of the International Conference on Machine Learning*, 2002.

Bengio, Yoshua, Courville, Aaron, and Vincent, Pascal. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8), 2013.

Bowman, S.R., Vilnis, L., Vinyals, O., Dai, A.M., Jozefowicz, R., and Bengio, S. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349*, 2015.

Burda, Yuri, Grosse, Roger, and Salakhutdinov, Ruslan. Importance Weighted Autoencoders. *arXiv preprint arXiv:1509.00519*, 2015a.

Burda, Yuri, Grosse, Roger, and Salakhutdinov, Ruslan. Accurate and conservative estimates of mrf log-likelihood using reverse annealing. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, 2015b.

Chen, Xi, Kingma, Diederik P., Salimans, Tim, Duan, Yan, Dhariwal, Prafulla, Schulman, John, Sutskever, Ilya, and Abbeel, Pieter. Variational Lossy Autoencoder. In *International Conference on Learning Representations*, 2017.

Dieleman, Sander, Schlter, Jan, Raffel, Colin, Olson, Eben, Sønderby, Søren K, Nouri, Daniel, van den Oord, Aaron, and and, Eric Battenberg. Lasagne: First release., August 2015.

Fraccaro, Marco, Sønderby, Søren Kaae, Paquet, Ulrich, and Winther, Ole. Sequential neural models with stochastic layers. In *Advances in Neural Information Processing Systems*. 2016.

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

Gulrajani, Ishaan, Kumar, Kundan, Ahmed, Faruk, Ali Taiga, Adrien, Visin, Francesco, Vazquez, David, and Courville, Aaron. PixelVAE: A latent variable model for natural images. *arXiv e-prints*, 1611.05013, November 2016.

Ioffe, Sergey and Szegedy, Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of International Conference on Machine Learning*, 2015.

Jang, Eric, Gu, Shixiang, and Poole, Ben. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

Kingma, Diederik and Ba, Jimmy. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 12 2014.

Kingma, Diederik P., Rezende, Danilo Jimenez, Mohamed, Shakir, and Welling, Max. Semi-Supervised Learning with Deep Generative Models. In *Proceedings of the International Conference on Machine Learning*, 2014.

Kingma, Diederik P, Salimans, Tim, Jozefowicz, Rafal, Chen, Xi, Sutskever, Ilya, and Welling, Max. Improved variational inference with inverse autoregressive flow. In *Advances in Neural Information Processing Systems*. 2016.

Kingma, Diederik P; Welling, Max. Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*, 12 2013.

Krishnan, Rahul G, Shalit, Uri, and Sontag, David. Deep Kalman filters. *arXiv:1511.05121*, 2015.

Lake, Brenden M, Salakhutdinov, Ruslan R, and Tenenbaum, Josh. One-shot learning by inverting a compositional causal process. In *Advances in Neural Information Processing Systems*. 2013.

Maaløe, Lars, Sønderby, Casper K., Sønderby, Søren K., and Winther, Ole. Auxiliary Deep Generative Models. In *Proceedings of the International Conference on Machine Learning*, 2016.

Maddison, Chris J., Mnih, Andriy, and Teh, Yee Whye. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, abs/1611.00712, 2016.

Miyato, Takeru, Maeda, Shin-ichi, Koyama, Masanori, Nakae, Ken, and Ishii, Shin. Distributional Smoothing with Virtual Adversarial Training. *arXiv preprint arXiv:1507.00677*, 7 2015.

Ranganath, Rajesh, Tran, Dustin, and Blei, David M. Hierarchical variational models. *arXiv preprint arXiv:1511.02386*, 11 2015.

Rasmus, Antti, Berglund, Mathias, Honkala, Mikko, Valpola, Harri, and Raiko, Tapani. Semi-supervised learning with ladder networks. In *Advances in Neural Information Processing Systems*, 2015.

Rezende, Danilo J., Mohamed, Shakir, and Wierstra, Daan. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. *arXiv preprint arXiv:1401.4082*, 04 2014.

Rezende, Danilo Jimenez and Mohamed, Shakir. Variational Inference with Normalizing Flows. In *Proceedings of the International Conference on Machine Learning*, 2015.

Rolfe, Jason Tyler. Discrete variational autoencoders. In *Proceedings of the International Conference on Learning Representations*, 2017.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. *arXiv preprint arXiv:1606.03498*, 2016.

Sønderby, Casper Kaae, Raiko, Tapani, Maaløe, Lars, Sønderby, Søren Kaae, and Winther, Ole. Ladder variational autoencoders. In *Advances in Neural Information Processing Systems 29*. 2016.

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

Tenenbaum, Joshua B., Griffiths, Thomas L., and Kemp, Charles. Theory-based Bayesian models of inductive learning and reasoning. *Trends in cognitive sciences*, 10 (7):309–318, July 2006.

Tran, Dustin, Ranganath, Rajesh, and Blei, David M. Variational Gaussian process. In *Proceedings of the International Conference on Learning Representations*, 2016.