Latent space conditioning for improved classification and anomaly detection

Norlander, Erik; Sopasakis, Alexandros

2019

Latent space conditioning for improved classification and anomaly detection

Erik Norlander and Alexandros Sopasakis

Lund University

November 26, 2019

Abstract

We propose a variational autoencoder to perform improved pre-processing for clustering and anomaly detection on data with a given label. Anomalies however are not known or labeled. We call our method conditioned variational autonencoder since it separates the latent space by conditioning on information within the data. The method fits one prior distribution to each class in the dataset, effectively expanding the prior distribution to include a Gaussian mixture model. Our approach is compared against the capabilities of a typical variational autoencoder by measuring their V-score during cluster formation with respect to the *k*-means and EM algorithms. For anomaly detection, we use a new metric composed of the mass-volume and excess-mass curves which can work in an unsupervised setting. We compare the results between established methods such as as isolation forest, local outlier factor and one-class support vector machine.

1 Introduction

Most anomaly and clustering methods will perform well on good data. Real life applications however do not follow any guidelines. Actual data is typically incomplete, faulty, missing, unbalanced or consist of outliers. Data can have a large amount of mixed categorical and numerical features making it difficult to ascertain their relational importance. Many times there exist a size imbalance between categories. Thus a machine learning model could bias towards the larger classes in the data set. Furthermore overlap can make it difficult to identify the class each of the data points belongs to. In this case, anomaly detection can be challenging since an anomalous data point for one class may be a normal point for another. Last but not least data could be wrongly labeled which in turn can lead to severe modeling errors.

Clustering is a classic machine learning problem that is well studied but it is also heavily data dependent [32]. The success of the clustering often comes down to feature engineering or otherwise pre-processing the data in some way if it's not already nicely placed in some euclidean space.

Anomaly detection is one of the most important problems in manufacturing [20], cyber security [28], medical imaging [37], fraud detection and several others. At the same time anomaly detection is a difficult problem since it is heavily depended on both data quantity and quality. The problem becomes even more complicated when considering an *unsupervised* setting such as we do here.

We consider data which is unmarked in terms of containing anomalies or not. As a result, while the class label is known, we do not know which points are anomalous and which are normal. For this case it's important to specify what we mean by *anomaly*. In this paper, we will consider anomalies to be data points which occur in low probability regions of the dataset. We use dimensionality reduction to find these probabilities and evaluate their degree of anomaly using the EM-MV measure [9]. More importantly however our proposed framework will be able to work with a broad category of data in an unsupervised manner. It will also allow us to use established methods of clustering and anomaly detection for comparison purposes.

1

We propose to use a variation of the VAE [14] for this purpose. Instead of using the generative aspects of the VAE, we will perform our analysis in *latent space*. This is a reduced dimensionality space which forces the data to be close to some prior distribution. The latent space will also retain relational information and make it possible to observe how data features compare with each other. We propose to name this algorithm *Conditional Latent Space Variational Autoencoder* or CL-VAE for short.

We apply two different clustering methods in the latent space: the k-means and the expectation maximization (EM). Specifically the EM algorithm [15] alternates between estimating the function of the expectation of the log-likelihood for each data point and then maximizing that expectation. This method therefore iteratively updates the model parameters and creates a probability for each point belonging to a cluster. The k-means method relies on Lloyd's algorithm [18] and works much like the EM method described above in order to cluster the data. The main difference from the EM method above is that we must provide the number of data clusters as an input. Overall however the algorithm is simple in its implementation. We outline the k-means method below in order to better highlight how clusters are created:

0. Assume k clusters are to be established from the data.

1. Randomly assign k points as centroids.

2. Assign each of the points in the dataset to belong to the closest of the k centroids.

3. Compute k new centroids of each cluster based on its assigned points from above.

4. Repeat steps 2 and 3 until centroids do not change or maximum iterations exceeded.

Finally we apply the V-score as a widely accepted measure of success or failure of each clustering method.

We begin in Section 2 with a brief overview of the clustering and anomaly detection metrics which we apply during this study. In Section 3 we present the theory behind VAE's and in particular how exactly we condition our VAE to fit multiple Gaussians and categorize the latent space based on the data. The choice of loss function is critical in forming the clusters. In Section 3.3 we provide the theoretical background on the Kullback-Leibler (KL) divergence used for regularization as well as the reconstruction piece of the loss function. We present the unsupervised case of anomaly detection in Section 4.2 where we also discuss the proposed EM-MV measure to evaluate its performance. We end with a number of remarks and an overview in Section 5.

2   Approach and state of the art

There are a number of approaches to anomaly detection using generative models; many, using the reconstruction error as a measure of "degree" of abnormality [1] [21]. However in this paper we implement a VAE as a pre-processing algorithm for use in common anomaly detection algorithms. In order to do so effectively, we introduce the CL-VAE which shapes the latent space by discovering which distribution each of the data belongs to and making it easier to work with.

The idea of using a Gaussian Mixture Model (GMM) as prior for training a VAE is not a new idea [4, 30]. However, in the case of GMVAE [4], there is no conditioning on a pre-defined label. Rather, it tries to estimate the prior distribution with Monte Carlo methods. A similar algorithm was cleverly described by J. Su [12] but it includes a classifier that we do not really need here since we know which class to assign to which Gaussian. This is also related to the DEC-algorithm by J. Xie et al. [36] and the VaDE-algorithm by Z. Jiang et al. [13] with a built-in unsupervised cluster assignment which is not something that we attempt here. Our algorithm is different from the CVAE (Conditional Variational Autoencoder) in [5,31] because we're conditioning the latent space itself on the class label by selecting the appropriate Gaussian, not assuming we can map all classes to the same Gaussian as CVAE

does. Instead the CVAE conditions the encoder and decoder, keeping the one-gaussian assumption in the latent space which we avoid.

Two different datasets will be used in this paper in order to illustrate different properties of the proposed methodology. First, the standard MNIST dataset will be used to present clustering and anomaly detection capabilities in order to compare with other publications and results on that dataset. The second dataset we use consists of actual trades made by traders at a Swedish investment bank. Thus this data set is typical of the many problems a real dataset can have: unbalanced classes, mislabeling, missing data etc. This dataset was also used in the thesis [22] and is included to diversify the discussion on anomaly detection as well as to illustrate a real application of the proposed approach.

We will first train and produce latent spaces from the two different autoencoders: VAE and CL-VAE. We then compare how well we can perform clustering and anomaly detection on each. For the sake of visualization we have chosen to present the results in 2 latent dimensions. We note however that performance could increase if we were to use more dimensions in latent space as it would include more information. Arguably two of the most standard clustering algorithms, which we also use in this paper, are the k-means [18] and EM [15] algorithms. Similarly, for anomaly detection the most frequently used algorithms in this area are Isolation Forest [17], LOF [2] and OCSVM [27].

2.1 Detection methodology

A typical issue for most clustering algorithms is how to evaluate their results. This is because, while clustering itself is an unsupervised task, in some cases we can check the clusters that do form against some ground truth. In this case, that would be class label.

The V-score or validity score is one such metric capable of measuring this kind of clustering result. It is an entropy (12) based approach and works by measuring two interdependent characteristics in clusters: *completeness* and *homogeneity*.

Homogeneity measures whether every member of a given cluster belongs to a single class. This is done as follows,

$$h = \begin{cases} 1, & \text{if} \quad H(C,K) = 0, \\ 1 - \frac{H(C|K)}{H(C)}, & \text{otherwise.} \end{cases}$$

Note that the case with cross entropy of $H(C|K) = 0$ denotes a totally homogeneous cluster.

Completeness on the other hand measures whether all known members of a class are assigned to the same cluster. We define completeness via,

$$c = \begin{cases} 1, & \text{if} \quad H(K,C) = 0, \\ 1 - \frac{H(K|C)}{H(K)}, & \text{otherwise.} \end{cases}$$

Note that $0 \leq c \leq 1$. In practice however we normalize the conditional entropy by $H(C)$ in order to remove any class size dependencies.

The validation score or V-score is defined to be the weighted mean between c and h,

$$V_\beta = \frac{(1+\beta)hc}{\beta h + c},$$

where β is a weight favoring completeness if $\beta > 1$ or homogeneity if $\beta < 1$.

Readers might recognize how the completeness and homogeneity defined above encompass metrics such as inertia and the Dunn index [7] which have been used in the past to measure intra-clustering and inter-clustering distances.

2.2 Anomaly detection

Confirming anomalies which are not already labeled is not a trivial task. In this unsupervised setting we can not use Precision-Recall or ROC simply because there's no way to check our results against a ground

truth. Therefore, the excess mass-mass volume (EM-MV) method can be a great tool at indicating anomalies as it has been found to agree with its supervised counterparts [9].

We assume that anomalies occur on the tail ends of a probability distribution. So our goal is to estimate the density level curves of that distribution. The method works on level sets for which a function f equals a given constant c, $L_c(f) = \{(x_1, \ldots, x_n) \mid f(x_1, \ldots, x_n) = c\}$. In our case the function f is the probability density which we estimated by our CL-VAE in latent space. The degree of abnormality or anomaly for the EM-MV method is given by a scoring function $s : \mathbb{R}^d \to \mathbb{R}_+$ for data in $\mathbb{R}^d$. The method relies on the mass volume (MV_s) and excess mass (EM_s) curves of s as follows,

$$MV_s(\alpha) = \inf_{u \geq 0} \quad \text{Leb}(s \geq u) \text{s.t.} \quad \mathbb{P}(s(X) \geq u) \geq \alpha, \tag{1}$$

$$EM_s(t) = \sup_{u \geq 0} \quad \{\mathbb{P}(s(X) \geq u) - t\text{Leb}(x \geq u)\}, \tag{2}$$

where $t > 0, \alpha \in (0,1)$ and $\mathbb{P}(s \geq t) = \frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{s(X_i) \geq t}$. So now we can evaluate our method by computing the distance between the level sets of f and s for excess mass and mass volume as follows: $||EM_s - EM_f||_{L^1(I)}$ and $||MV_s - MV_f||_{L^1(J)}$ [9]. Practically we let $I = [0.9, 0.999]$ and $J = [0, EM_s^{-1}(0.9)]$ where $EM_s^{-1}(0.9) = \inf\{t \geq 0, EM_s(t) \leq 0.9\}$ as shown in [9]. One technical issue involves the computation of the Lebesgue terms above which can be resolved via Monte Carlo estimation.

The measure is then based on the area under the EM_s and MV_s curves. This area should be *maximized* for EM_s and *minimized* for MV_s.

3 Creating Generative Models

As mentioned earlier, Variational Autoencoders (VAEs) are generative models introduced in 2014 by Kingma and Welling [14]. The purpose of such a model is density estimation. Meaning, that a VAE can estimate any underlying distribution in a dataset and start to generate new samples from that dataset.

The VAE consists of a encoder-decoder pair of neural networks, with a stochastic latent layer in the middle as depicted in Figure 1. The encoder which we express here with $p_\theta(z|x)$ processes the input x and produces the parameters of a Gaussian distribution represented by z in the latent layer as shown in Figure 1. The decoder which we denote by $p_\theta(\hat{x}|z)$ uses that Gaussian distribution z in latent space as input to produce the parameters to an approximation of the probability distribution describing the original data. The weights and biases of the encoder and decoder networks are represented in θ.
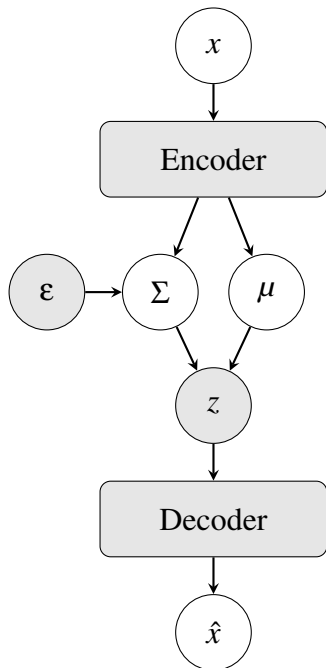
Figure 1: Diagram of a VAE structure. The encoder neural network maps the input x to the latent space described by the Gaussian distribution z. Then the Decoder neural network creates a reconstruction of the input in x̂.

VAEs allow us to effectively compress information within the data through dimensionality reduction occurring in the latent space. Since the decoder goes from a smaller to a larger space information is lost. We measure this loss by the reconstruction loss which is the log-likelihood of $p_\theta(\hat{x}|z)$ which will give us a loss function to minimize in training. This shows also how effective the encoder was in compressing information within z from the original data x. Furthermore VAEs can also be used with categorical data as well as non-linear type transformations in contrast to PCA type methods [23].

Note that the VAE architecture is very similar to that of an autoencoder shown in Figure 10 in the Appendix except for the latent space which is sampled by estimating μ and Σ. This is done by mapping the mean and variance of these points with the stochastic layer described by ε, μ and Σ. We also note that every point in the latent space of a VAE is forced to have feasible features since it is assigned a variance. A prime advantage of this approach is that VAEs, unlike regular autoencoders, map values to the latent space by retaining relational information between them so that they remain meaningful for analysis purposes [3].

Another type of generative model is a GAN [10] that solves the same problem with a game-theoretic approach. The VAE on the other hand is derived from a purely statistical point of view and the fact that you can use if for generation is almost just a bonus [35]. Because of the statistical interpretability and ease of manipulation, we chose to use a VAE for this task.

3.1 The optimization function

The choice of loss function during any optimization or neural network problem is critical and has direct implications in success or failure of the given problem. To create a proper loss function however requires knowledge of $p_\theta(x)$ in order to compute the $\log p_\theta(\hat{x}|z)$ which is intractable (see Appendix B). To counter this problem we define a lower bound of the likelihood and optimize that instead.

Assuming that z can be estimated using the distribution $q_\phi(z|x)$ we work on estimating the logarithm of $p_\theta(x)$ (see also (9) in Appendix B),

$$\log p_\theta(x) = \log \mathbb{E}_z \left[p_\theta(x|z) \right] = \mathbb{E}_{z \sim q_\phi(z|x)} \left[\log p_\theta(x) \right]$$

$$= \mathbb{E}_z \left[\log p_\theta(x|z) \right] - \mathbb{E}_z \left[\log \frac{q_\phi(z|x)}{p_\theta(z)} \right] + \mathbb{E}_z \left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)} \right].$$

This however is a typical application for the Kullbag-Leibler (KL) divergence (see Appengix C or [33]). The Kullback-Leibler divergence is a semi-metric [33] and can provide an estimate of the distance between two measures P and Q,

$$D_{KL}(P||Q) = \mathbb{E}_{x \sim P} \left[\log \frac{P(x)}{Q(x)} \right].$$

Using the KL divergence therefore we can express $\log p_\theta(x)$ as,

$$\log p_\theta(x) = \mathbb{E}_{z \sim q_\phi(z|x)} \left[\log p_\theta(x|z) \right] - D_{KL}(q_\phi(z|x)||p_\theta(z)) + D_{KL}(q_\phi(z|x)||p_\theta(z|x)). \quad (3)$$

The first term above is possible to compute directly from the decoder and sampling with the reparameterization trick [14], which will be discussed later. The second term is a KL-divergence between a general and a normal Gaussian which has a closed form solution based on Property 2 of Appendix C. Finally, the problematic last term can be shown to be greater than 0 [34] through Property 1 in Appendix C. Therefore we can define the lower bound of (3) as,

$$L(\theta, \phi; x) = \mathbb{E}_{z \sim q_\phi(z|x)} \left[\log p_\theta(x|z) \right] - D_{KL}(q_\phi(z|x)||p_\theta(z)). \quad (4)$$

Thus our task is to minimize the above with respect to the parameters θ and ϕ,

$$\phi^*, \theta^* = \underset{\phi, \theta}{\operatorname{argmin}} L(\theta, \phi; x). \quad (5)$$

The only remaining issue is to resolve the expected value in (4). This can be accomplished with a reparameterization idea from [14] provided below for completeness.

3.2 The change of variables idea

Functions such as (4) are not uncommon in optimization problems [11]. The novel idea, introduced in [14], which removes difficulties of the expected value in (4) is to introduce a suitable change of variables.

We express $z \sim q_\phi(z|x)$ through a deterministic transformation g_ϕ and a random variable $\varepsilon \sim p(\varepsilon)$ as $z = g_\phi(x, \varepsilon)$, where $p(\varepsilon)$ is a simple distribution which does not depend on x or ϕ. In our case we choose $p(\varepsilon) = N(0,1)$ since we want the latent space to be Gaussian. This reduces z to,

$$z = g_\phi(x, \varepsilon) = \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon, \quad \varepsilon \sim N(0,1)$$

We now define an auxiliary function $f(x,z) = \log p_\theta(x,z) - \log q_\phi(z|x)$ over the distribution $q_\phi(z|x)$ and write (4) as an expectation of f. If we then substitute z and compute the gradient with respect to ϕ we get,

$$\nabla_\phi \mathbb{E}_{z \sim q_\phi(z|x)}[f(x,z)] = \nabla_\phi \mathbb{E}_{g_\phi(x,\varepsilon) \sim p(\varepsilon)}[f(x, g_\phi(x,\varepsilon))] = \mathbb{E}_{g_\phi(x,\varepsilon) \sim p(\varepsilon)}[\nabla_\phi f(x, g_\phi(x,\varepsilon))].$$

The expectation and the gradient commute which allows us to practically optimize the above using back propagation.

3.3 The Conditional Variational Autoencoder

VAEs are designed to estimate the unknown probability distribution $p_\theta(x)$ for the given x using a Gaussian in latent space. Instead we propose to use information which is already available in the data in order to more accurately express the dataset x through several Gaussians. We specify the Gaussians in latent space by conditioning on any one of the labels in the data set. For the MNIST data set of images for instance we condition on the number label associated with each image. Therefore for $N = 10$ categories in that label in the data,

$$p_\theta(x) = \sum_{y=1}^{N} \int p_\theta(x|y,z) p_\theta(z|y) p_\theta(y) dz. \tag{6}$$

where we chose $p_\theta(y)$ to be a weighted uniform distribution since the numbers in the MNIST data are discrete constants and $p_\theta(z|y)$ is a set of N Gaussians each with unknown mean and variance. Finally to understand $p_\theta(z|x,y)$ we apply Bayes rule,

$$p_\theta(z|x,y) = \frac{p_\theta(x|y,z) p_\theta(z|y) p_\theta(y)}{p_\theta(x)}, \tag{7}$$

where $p_\theta(z|x,y)$ can be estimated from the encoder $q_\phi(z|x,y)$. Note furthermore that $p_\theta(y)$ is known and gives $p_\theta(z|y)$. We can therefore write $p_\theta(x|y,z) = p_\theta(x|z)$.

Following arguments similar to constructing the loss function for VAE we now produce the following loss function for our CL-VAE,

$$L(\theta,\phi;x) = \mathbb{E}_{z \sim q_\phi(z|x)} \Big[\log p_\theta(x|z) \Big] - D_{KL}(q_\phi(z|x,y)||p_\theta(z|y)). \tag{8}$$

Since y is discrete the KL divergence above can be written as,

$$D_{KL}(q_\phi(z|x,y)||p_\theta(z|y)) = \mathbb{E}_{z|y} \left[\log \frac{q_\phi(z|x,y)}{p_\theta(z|y)} \right] = \sum_{y=1}^{N} p_\theta(z|y) \log \frac{q_\phi(z|x)}{p_\theta(z|y)}$$

To produce a closed form solution for $D_{KL}(q_\phi(z|x,y)||p_\theta(z|y))$ in the non-normal Gaussian case we follow ideas from [12] and let $q_\phi(z|x)$ be some Gaussian distribution that is conditioned on x,

$$q_\phi(z|x) = \frac{\exp\left\{ -\frac{1}{2} \left\| \frac{z - \mu(x)}{\sigma(x)} \right\|^2 \right\}}{\prod_{i=1}^{d} \sqrt{2\pi\sigma_i^2(x)}}$$

We now let $p_\theta(z|y)$ be Gaussian with mean μ_y and variance 1,

$$p_\theta(z|y) = \frac{1}{(2\pi)^{d/2}} \exp\left\{ -\frac{1}{2} ||z - \mu_y||^2 \right\}$$

and use in simplifying,

$$\log \frac{q_\phi(z|x)}{p_\theta(z|y)} = -\frac{1}{2} \sum_{i=1}^{d} \log \sigma_i^2(x) - \frac{1}{2} \left\| \frac{z - \mu(x)}{\sigma(x)} \right\|^2 + \frac{1}{2} ||z - \mu_y||^2.$$

We apply the same reparameterization idea as before with $z = g_\phi(x,\varepsilon) = \mu_\phi(x) + \sigma_\phi(x) \odot \varepsilon$ where $\varepsilon \sim N(0,1)$ which simplifies the expression above to,

$$\log \frac{q_\phi(z|x)}{p_\theta(z|y)} = -\frac{1}{2} \sum_{i=1}^{d} \log \sigma_i^2(x) + \frac{1}{2} ||z - \mu_y||^2.$$

Computation of $\sum_{y=1}^{N} p(z|y) \log \frac{q_\phi(z|x)}{p_\theta(z|y)}$ now reduces to a simple matrix operation.
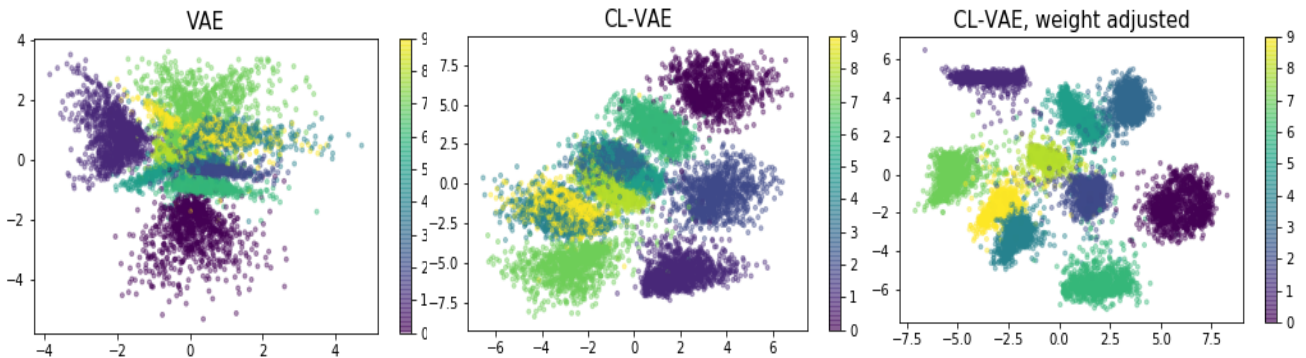
4 Clustering and Anomaly Detection Results

Figure 2: The 2D-latent space for the MNIST test data. Here the colors represent the 10 number classes for the VAE (left), the CL-VAE (middle) and a weight adjusted CL-VAE (right). The CL-VAE produces a clear latent space fitting and automatic separation between classes.

Training a regular VAE on the MNIST dataset we are treated by a latent space representation where the classes accumulate around a single Gaussian. The model is trained on minimizing the loss function (4) with a single Gaussian normal prior. The result of this is clear in the top of Figure 2 as the points are all expanding outward from the origin where the normal Gaussian has its mean. As can be seen in that figure, there is some structure and class separation but clusters have inconsistent shapes and are overlapping heavily. Class overlap is natural since numbers can easily resemble each other and can be hard to tell apart for humans as well. However, class overlap will remain a problem when using the VAE. This is attributed to the fact that classes are trying to adapt their form to a single Gaussian prior.

We also train the CL-VAE on the same data set and present the results in the middle of Figure 2. It is immediately evident, at least visually, that the class separation has now improved compared with that from a regular VAE. The class overlap is still visible since numbers due to their inherent shape resemble each other in the MNIST data. In that respect it is not surprising to see that the distribution for numbers 6 and 0 are close to each other and the clusters forming 4 and 9 are completely overlapping. We also see that 4 and 9 seem to be completely overlapping and 3, 5 and 7 are partially overlapping. This is not going to be helpful for clustering.

Finally, we also show a weight adjusted CL-VAE in the bottom of Figure 2. Here the reconstruction error is weighted down s.t. $\gamma = 0.3$ and $\beta = 1.0$. This means that more focus is put on the KL-divergence term in training. Visually, we can see that 4 and 9 no longer overlap and no other class is completely overlapping with any other.

4.1 Clustering

We now present and analyze a number of clustering results comparing our proposed CL-VAE and a regular VAE. During our analysis we use two different clustering methods: the k-means and the expectation maximization (EM). A visual inspection of the results is always helpful although we also measure the V-score in order to better assert the success of each algorithm.
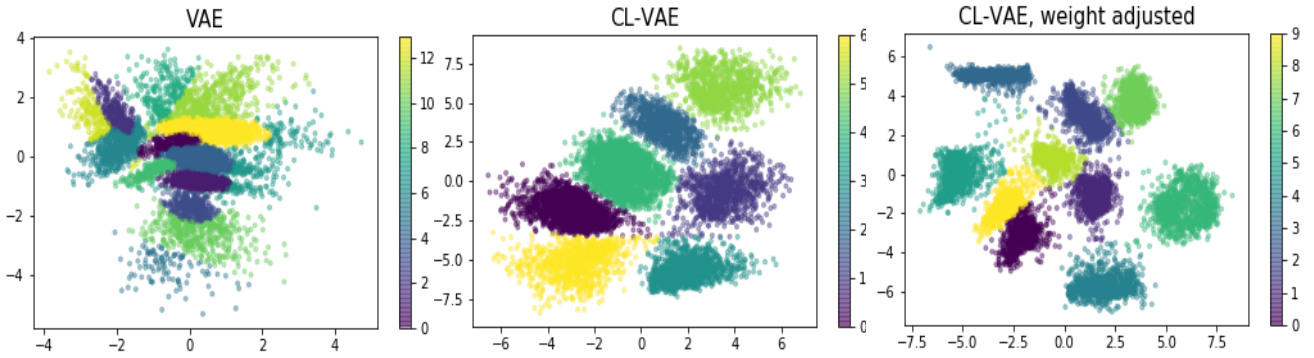
Figure 3: The results of performing EM-clustering on the three different latent spaces of Figure 2 using the optimal number of clusters from Table 1. Comparing this Figure with Figure 2 we can see which numbers ended up in each of the clusters. Note that the numbers 4 and 9 ended up in the same cluster for the standard CL-VAE.

Based on the latent space representations from VAE and CL-VAE we compute their respective V-scores in order to find the optimal number of clusters for each. For the VAE, the V-score is quite bad for both the EM and k-means clustering algorithms as can be seen in Table 1. Another problem is that it reaches its maximum V-score using 14 clusters, more than the true amount of numbers in the dataset. This however is not that strange since the clusters based on the VAE latent space shown in Figure 2 are not clearly separated.

Table 1: The results of clustering for each of the 3 different latent spaces. The left column under type of clustering algorithm is the optimal V-score and the right column is the optimal number of clusters. As is evident, the CL-VAE with the reconstruction error weighted down performs the best (boldface).

Algorithm	EM		k-means	
VAE	0.6313	14	0.5572	14
CL-VAE	0.8132	7	0.8088	7
$\gamma = 0.3$	**0.9248**	**10**	0.9233	10

For the CL-VAE we get a higher V-score for both the EM and k-means algorithms but the clusters are now too few instead. Inspecting Figure 3 and comparing it with Figure 2 we see that the numbers 4 and 9 ended up in the same cluster. The same is true for the numbers 3, 5 and 7 as well.

Finally, considering the weight adjusted CL-VAE we find that we get 10 unique clusters and again, a much higher V-score for both EM and k-means. Using this setup you can therefore actually cluster the numbers quite successfully. As a result we will use the weight adjusted CL-VAE to identify anomalies in the next section.

4.2 Anomaly detection

In this section we present and compare unsupervised anomaly detection capabilities based on the latent space of the proposed CL-VAE versus that of the regular VAE. We perform these comparisons under the context of three of the most widely used methods for anomaly detection. We also note that neither of our data sets include information or labels as to whether a given data point is an anomaly or not.

Some of the algorithms applied here to identify anomalies include hyper-parameters which must be tuned. We have therefore undertaken this task and in all results presented we have already established the best set of such hyper-parameters for those methods.

Table 2: Comparisons of excess mass (EM_s) and mass volume (MV_s) scores for three different anomaly detection methods based on either the VAE or the CL VAE latent space of MNIST. The results indicate that the one-class support vector machine (OCSVM) performs best for both latent spaces. However, during further investigation OCSVM was ruled out as it classifies about half of the data as anomalous. Note that the latent space of the VAE gets better scores than the CL-VAE with all three algorithms.

LS/Algo.	EM_s (10^{-3})	MV_s
VAE		
IF	2.532	2.465
LOF	2.014	3.136
OCSVM	**2.533**	**2.442**
CL-VAE		
IF	0.884	7.065
LOF	0.841	7.236
OCSVM	**0.926**	**6.754**

Remembering that we should maximize the EM_s score and minimize the MV_s score, the results presented in Table 2 clearly indicate that the VAE produces better EM-MV scores than the CL-VAE. This is rather unexpected and something we'll discuss further. It's also clear that the OCSVM provides the best EM-MV scores for both types of latent spaces. Investigating a little further, we find that the OCSVM classifies more than half of the test data as anomalous. This obviously deems it rather useless.

Ruling out the OCSVM, the Isolation Forest algorithm performs the best on both latent spaces. Closer investigation of the anomaly classification mechanism however reveals that the IF algorithm classifies the center of the prior distribution for the VAE as normal and the edges of the distribution as anomalous. This means that, while the number of elements in each class are approximately the same, the numbers 0, 1 and 7 are over-represented among the anomalies while numbers 6, 9, 2, 8 and 5 are under-represented. Taking a look at the top Figure 2 again the reason for this becomes clear as there is plenty of class overlap in the center of the distribution for the VAE where 6, 9, 2, 8 and 5 are all put in the middle. At the same time, 0, 1 and 7 occupy the fringes of the distribution, leading to the over-representation in the number of anomalies.

This behavior is not as prevalent in the LOF algorithm which actually finds some anomalies between clusters, leading to a more well-balanced distribution of anomalies among the different classes. So even though the LOF performed the worst on the EM-MV score, it seems to be the most reasonable classifier of anomalies in this case.
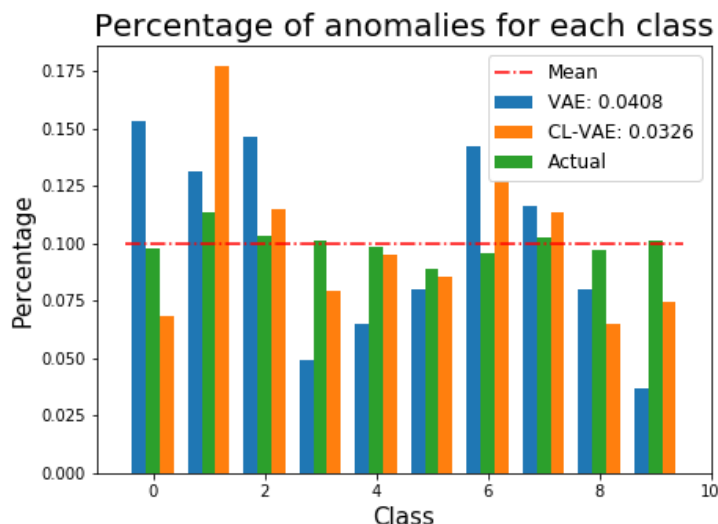
Figure 4: A bar chart comparing the anomalies (in percent) found within each class when performing anomaly detection on the two different latent spaces. The CL-VAE gets closer to the actual distribution and therefore gets a lower RMSE, which is the number displayed in the box next to each algorithm.

Now considering the weight adjusted CL-VAE, we find that again disregarding the OCSVM, Isolation Forest gets the best EM-MV scores. However, the same over- and under-representation problems apply here. It seems like it weighs the regions with classes that are slightly overlapping as 'more normal' while classes that are clearly distinct are considered 'more anomalous'. For instance, it finds 144 anomalies for the number 6's while only 14 anomalies for the number 8. Once again however LOF manages to provide us with a much more even distribution of anomalies.
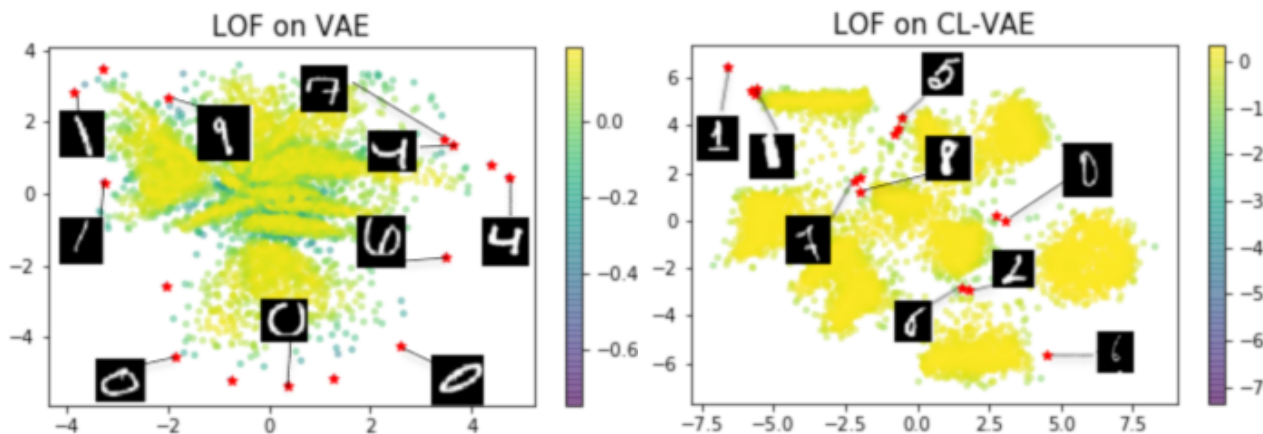

Figure 5: The results of running anomaly detection using the LOF algorithm on the two different latent spaces. In this case, the 15 most anomalous points are highlighted with a red star and some are filled in with their corresponding image in MNIST. As is clear, the VAE tends to overemphasize certain numbers in anomaly detection as was described earlier, that tend to end up on the fringes of the prior distribution. While the CL-VAE finds it's most uncommon anomalies between clusters that tend to be more unclear of what class they actually belong to.

Overall however we see that in general VAE produces better EM-MV scores than CL-VAE. This could be because the VAE latent space has a few points that end up far out on the tail of the distribution and therefore the algorithms have an easier time distinguishing normal points from anomalous. That being said, these points are mostly consisting of a few classes, leading to a more unbalanced set of anomalies which is clearly wrong. That is where the CL-VAE has an advantage. To visualize this, we have plotted the percentages of anomalies for each class in Figure 4. What is clear from this plot is that both autoencoders over-estimate and under-estimate certain classes. But overall, the CL-VAE get's a lower RMSE from the actual distribution of the dataset at 0.0326 compared with 0.0408 for the VAE.

To summarize our findings in this section we found that the VAE separates the data in a more extreme way than CL-VAE while at the same time overestimating anomalies for some classes at the edges of its single-blob type latent space. This higher degree of separation seems to be why the VAE achieves better EM-MV scores. The CL-VAE produces a more balanced set of anomalies in the dataset which stem for all of the classes in the data. Considering Figure 5, the most extreme anomalies seem more difficult to categorize than those anomalies identifies by the VAE.

4.3   Anomaly Detection in Trading Data

The dataset of trades from [22] contains about 100 000 rows and has columns describing what type of trade occurred. These include margin, nominal value, currency, type of instrument, counter party, portfolio and importantly *trader name*. By conditioning on what trader did what trade, we can try and create similar clusters to Figure 2. Instead of variations of handwritten numbers, the clusters describe the trading behavior of each trader in the dataset. Two overlapping traders for instance suggest that they traded in a similar fashion, meaning we have found a trading category. For more details related to the data we refer to the thesis work in [22].

Table 3: Comparisons of excess mass (EM_s) and mass volume (MV_s) scores for three different anomaly detection methods based on either the VAE or the CL-VAE latent space using the trading data. The results indicate that OCSVM performs best for VAE while Isolation Forest performs best for the CL-VAE. We note that all algorithms performed better on the latent space of the CL-VAE giving merit to our proposed method.

LS/Algo.	EM_s (e^{-3})	MV_s
VAE		
IF	4.648	1.362
LOF	3.587	1.736
OCSVM	**5.903**	**1.053**
CL-VAE		
IF	**16.33**	**0.5168**
LOF	13.48	0.5509
OCSVM	12.43	0.567

As can be seen in Table 3 we see that, on this dataset, the CL-VAE latent space outperforms the VAE latent space on the EM-MV measure for all algorithms tried. One big difference here is that the clusters that was formed with CL-VAE are placed much further apart than the clusters in MNIST as can be seen in Figure 6. This is likely contributing to the higher EM-MV score as it seems easier for the anomaly detection algorithms to contrast normal from anomalous points in such a latent space.
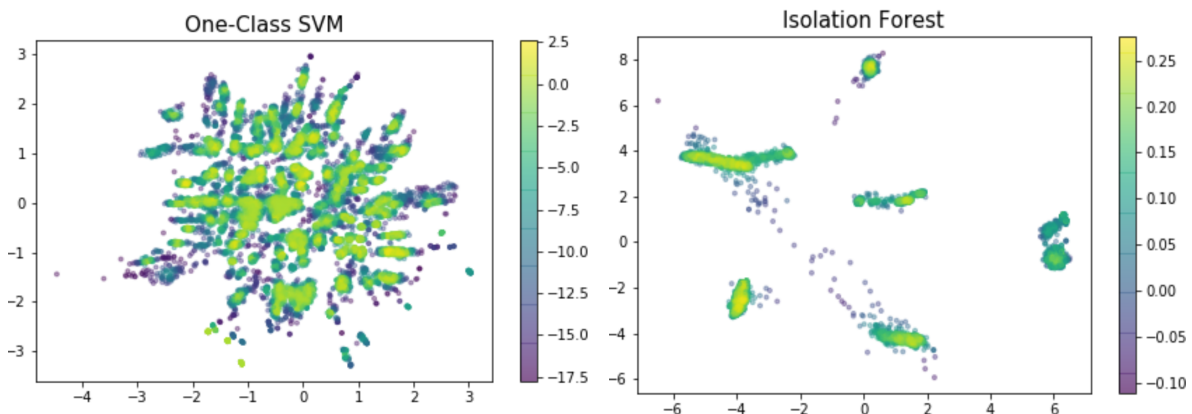

Figure 6: Comparing the two different latent spaces of the VAE (top) and the CL-VAE (bottom). The color coding corresponds to the best performing anomaly detection algorithm on each latent space.

4.4 Using Misclassification for Anomaly Detection

We now explore an alternative way toward anomaly detection. If a cluster is found in latent space where one most frequent class can be identified, then the points inside this cluster which do not belong to the most frequent class can be considered anomalies. To identify these points however can be tricky. These are points that in some sense behave more like the predicted class than its own class in the conditioned space, meaning that they should be classified as anomalies.

We will employ the V-score which is related to the Precision-Recall curves and the F1-score described in Section 4.1 to assist us in scoring and detecting these anomalies. As we showed in Table 1, the CL-VAE has a much better V-score than the VAE. In the case of the MNIST data this means that numbers which are classified to belong to a cluster will be moer likely to have the same label (homogeneity) and all of the members of that class are assigned to the same cluster (completeness).

We provide a representative sample of anomaly misclassification based on both the VAE and CL VAE latent spaces in Figure 7. As can be seen in that figure some of the handwritten numbers which were classified as 'anomalies' using the VAE latent space are rather easy to identify. Some would probably be considered anomalies but certainly not most. This is not the case for the CL-VAE where many of the found anomalies are in fact unintelligible.

Figure 7: A sample of misclassified numbers for each latent space. For the VAE many normal looking numbers were given the wrong label. Note also that mostly 3s, 7s and 9s were misclassified. Meanwhile, the misclassifications of the CL-VAE are much closer to the predicted label. Furthermore all labels are now included.

One of the big points about the CL-VAE is that points that end up on the tails of their respective prior distributions should be 'more anomalous'. There's no such guarantee with the VAE, because it only has one prior distribution. Instead, giving each class their own prior distribution should mean that the classes don't have to fight over placement in the latent space. To illustrate this further, we coloured the latent spaces according to the average deviation for each point compared with the respective mean of each class in Figure 8.
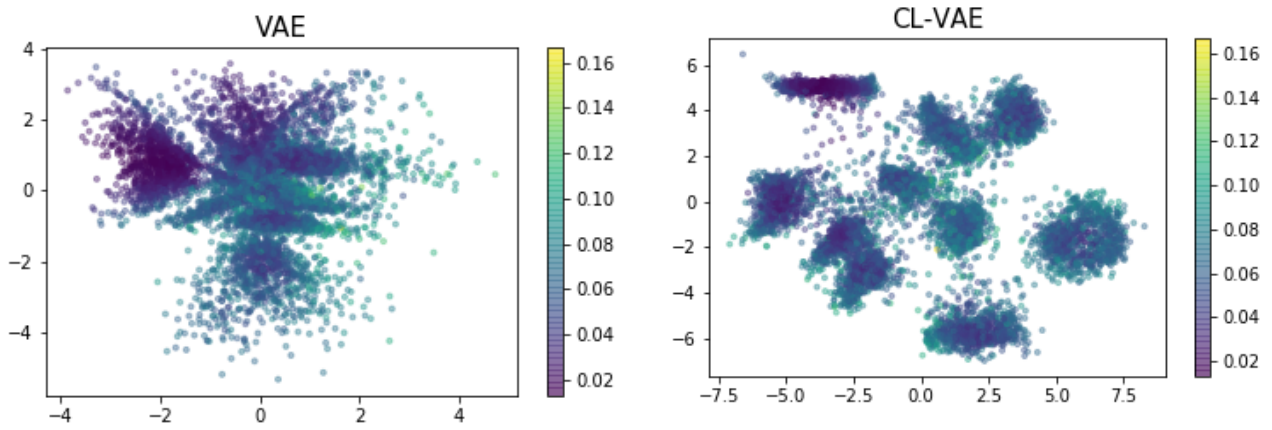
13

Figure 8: The latent space of the VAE (top) and the CL-VAE (bottom) colored with the RMSE of the numbers with respect to their means.

What this means is that points that are closer to the mean of each class also end up closer to their respective means in the latent space of the CL-VAE. If we compare it with the VAE, we see that the middle of the prior distribution actually tends to have a higher error than on some pockets on the tail of the distribution. Ultimately, what this means is that the latent space of the VAE can't be interpreted as samples from a Gaussian Normal distribution. Because there is no clear structure of where the points should end up based on their 'degree of abnormality'. This is in contrast with the CL-VAE, where the points can be interpreted as being sampled from a GMM prior. Consequently, points that either end up on the tail of their respective prior or end up in the wrong cluster, can in fact be called anomalies.

5   Discussion

In this article we explore classification of labeled data by conditioning in latent space. Conditioning allows us to use information within the data to improve clustering. We subsequently use these clusters to identify anomalies in the data. To showcase our findings we used the MNIST data set as well as actual trades from traders in a Swedish bank.

Our overall strategy for classification and anomaly detection involves a number of methodologies. We begin with automatic formation of clusters in latent space. These clusters are found and shaped according to optimal recommendations from our CL-VAE. In that respect we fit different Gaussians around each cluster via the EM clustering method. This improved clustering description allows us to apply a number of methods for detecting anomalies. Anomaly detection for us relies on detecting outliers in the latent space of the autoencoder. We compare among the k-means, the p-percentile, one-class SVM and LOF in order to identify the outliers for each of those clusters. Finally we apply multiple isolation forest (an isolation forest per cluster) in order to single out these anomalies.

We also note that neither of the datasets used was labeled in terms of whether a given point is an anomaly or not. This led us to propose methodologies which while unsupervised can indicate whether a given trade is typical or anomalous. The proposed EM-MV measure is able to detect anomaly candidates and the one-class SVM can indicate their severity. Furthermore the proposed methodology works with data of categorical nature (non-continuous) in order to find meaningful latent representations - both of which are not possible for classic PCA methods [23].

The proposed CL-VAE makes the latent space easier to understand while clustering data and performing anomaly detection. The question of how much of the proposed methodology is truly unsupervised should also be addressed. Clearly we are using labeling in the data to help us in the *initial* classification. So this is not an entirely unsupervised method. The data is forced into Gaussians based on that information. For the MNIST data for instance that corresponds to 10 clusters. However the methodology proposed is not completely supervised either since after the conditioning is performed the

method performs all actions in an unsupervised way based on a newly formed latent space. We do not supervise the algorithm in terms of how to find the anomalies after that. So this is still a form of unsupervised learning [3, 8].

We have shown that a weight adjusted CL-VAE can be a successful pre-processing procedure for clustering and eventual anomaly detection. It strongly outperformed the VAE on the clustering task for instance. While the VAE gets good EM-MV scores on the MNIST dataset, it does so while overestimating some classes and underestimating others. This problem is decreased when using the CL-VAE. Also, the anomalies that gets the worst possible scores with the CL-VAE do look much more unintelligible than the ones found for the VAE. Meanwhile, using the trading dataset, the CL-VAE obtained better EM-MV scores than the VAE.

The proposed CL-VAE attempts to make the latent space more understandable and suitable for analysis with established methods. It seems to succeed in this regard, as it both divides the latent space up in accordance with class labels and ensures that points that are improbable do in fact end up on the tail of their respective prior distributions or in other clusters.

It would be interesting to study the generative aspects of the CL-VAE as well in the future. In particular, it would be interesting to compare it with a CVAE, as they both have the functionality of generating samples from a given class, something that the VAE does not.

Appendix A. Background on Autoencoders

An autoencoder is a neural network capable of performing unsupervised dimensionality reduction. As a result it is able to discover a lower-level representation of a higher dimensional data space.

An autoencoder is typically constructed of two neural networks: an encoder and a decoder. An idealized autonencoder representation is given in Figure 9 where we reduce a 3 dimensional input to a 2 dimensional latent space.
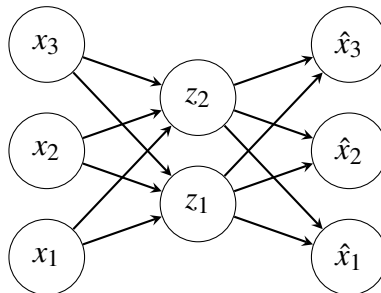

Figure 9: An idealization of an autoencoder. Here x_i is the input, z_i the hidden representation with 2 latent dimensions and $\hat{x}_i$ the reconstruction of the input x_i.

In general autoencoder applications both the encoder and the decoder networks are densely connected.

Definition 1 *Autoencoder. Given an input $x \in \mathbb{R}^d$ we assume that there is a mapping E to $z \in \mathbb{R}^p$ s.t. $E : x \rightarrow z$. This mapping is called the encoder and can be defined with an activation function σ, a weight matrix W and a bias term b as,*

$$z = \sigma(Wx + b).$$

Conversely, we assume that there is a mapping D s.t. $D : z \rightarrow x$ which is called the decoder and can be defined in a similar way to the encoder with the corresponding terms $\hat{\sigma}$, $\hat{W}$ and $\hat{b}$ as,

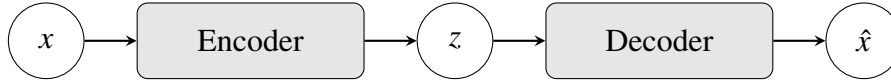$$\hat{x} = \hat{\sigma}(\hat{W}z + \hat{b}).$$

Figure 10: A flow chart of the autoencoder. Here the white circles represent input or output data and the grey rectangles represent neural networks.

According to this definition therefore the autoencoder's job is to estimate a non-linear transformation from x to z and its inverse from z to $\hat{x}$. In order for this autoencoder to compute these transformations it minimizes the reconstruction error of x,

$$L(x, \hat{x}) = ||x - \hat{x}||^2 = ||x - \hat{\sigma}(\hat{W}z + \hat{b})||^2 = ||x - \hat{\sigma}(\hat{W}(\sigma(Wx + b)) + \hat{b})||^2.$$

In general however autoencoders produce latent spaces which are not useful for analysis [3]. This is one of the main reasons that variational autoencoders have gained such popularity.

Appendix B. Variational autoencoders and Bayes rule

Variational autoencoders (VAEs) are able to discover the distributions responsible for the provided data. A VAE therefore solves the problem of probability density estimation and is a true generative model. This practically means that you can generate new samples from an unknown distribution [16]. Applications in image processing for example use VAEs to generate new images which retain some of the main features of the original data set [35].

If we have some set of locally observed variables x and we assume that they follow some unknown stochastic process X that we want to sample from, we can use some prior z that we assume to be Gaussian. By then taking the expectation of the conditional distribution of x given z, under z, we get the distribution for x from,

$$p_\theta(x) = \mathbb{E}_z\Big[p_\theta(x|z)\Big] = \int p_\theta(x|z) p_\theta(z) dz. \tag{9}$$

Note however that the integral above is intractable [14]. This is where Bayes rule can help.

We assume that the density functions $p(x)$ and $p(y)$ for stochastic variables x and y are known. If the conditional density function $p(x|y)$ is given then the conditional density function $p(y|x)$ can be computed from,

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)}. \tag{10}$$

Bayer rule will be useful in terms of computing the posterior $p_\theta(z|x)$,

$$p_\theta(z|x) = \frac{p_\theta(x|z)p_\theta(z)}{p_\theta(x)}. \tag{11}$$

However $p_\theta(x)$ above is not possible to compute in general. Instead we estimate the posterior distribution $p_\theta(z|x)$ using some other distribution $q_\phi(z|x)$ where ϕ contains estimates of the model parameters.

Appendix C. Kullbag-Leibler divergence

The KL-divergence has a number of imporant properties which we outline below. First we provide some definitions from information theory.

Given two probability distributions Q and P the entropy of P is defined from,

$$H(P) = \mathbb{E}_{x \sim P}[-\log P(x)], \tag{12}$$

and the cross-entropy of Q and P is given by,

$$H(P, Q) = \mathbb{E}_{x \sim P}[-\log Q(x)]. \tag{13}$$

Given two probability distributions Q and P we define the Kullback-Leibler divergence by taking the cross-entropy minus the entropy,

$$D_{KL}(P||Q) = H(P,Q) - H(P) = \mathbb{E}_{x \sim P}\left[\log \frac{P(x)}{Q(x)}\right]. \tag{14}$$

The Kullback-Leibler divergence measures how well Q approximates P.

Property 1 *Properties of KL-divergence [6].*

1. *if $P = Q$ then $D_{KL}(P||Q) = 0$,*

2. *if $P \neq Q$ then $D_{KL}(P||Q) > 0$.*

Property 2 *Solution to $D_{KL}(q(x)||p(x))$ in the normal Gaussian case [14]. Let's assume that x is some random variable in $q(x) \sim N(\mu,\sigma)$ and $p(x) \sim N(0,I)$*

$$D_{KL}(q(x)||p(x)) = \mathbb{E}_{z \sim q_\phi(z)}\left[\log \frac{q(x)}{p(x)}\right] == \int \log \frac{q(x)}{p(x)} q(x) dx == \int (\log q(x) - \log p(x)) q(x) dx.$$

This can now be evaluated as two different integrals. The first being

$$\int q(x) \log q(x) dx == \int N(x;\mu,\sigma^2) \log N(x;\mu,\sigma^2) dx == \frac{J}{2} \log(2\pi) + \frac{1}{2} \sum_{j=1}^{J}(1 + \log \sigma_j^2),$$

and the second,

$$\int q(x) \log p(x) dx == \int N(x;\mu,\sigma^2) \log N(x;0,I) dx == \frac{J}{2} \log(2\pi) + \frac{1}{2} \sum_{j=1}^{J}(\mu_j^2 + \sigma_j^2),$$

giving us the closed form solution,

$$D_{KL}(q(x)||p(x)) == -\frac{1}{2} \sum_{j=1}^{J} 1 + \log \sigma_j^2 - \mu_j^2 - \sigma_j^2.$$

References

[1] A. Borghesi, A. Bartolini, M. Lombardi, M. Milano and L. Benini, Anomaly Detection using Autoencoders in High Performance Computing Systems, (2018), Available at: *https://arxiv.org/pdf/1811.05269.pdf*

[2] M. M. Breunig, H. Kriegel, R. Ng and J. Sander, : LOF: Identifying Density-Based Local Outliers, in Proc. ACM Sigmod Int. Conf. on Management of Data, Dallas, TX, (2000).

[3] F. Chollet, Deep Learning With Python, Manning Publ., (2017).

[4] N. Dilokthanakul1, P. A. M. Mediano, M. Garnelo, M. C. H. Lee, H. Salimbeni, K. Arulkumaran and M. Shanahan, Deep Unsupervised Clustering with Gaussian Mixture Variational Autoencoders, (2016), Available at: *https://arxiv.org/pdf/1611.02648.pdf*

[5] C. Doersch, Tutorial on Variational Autoencoders, (2016), Available at: *https://arxiv.org/pdf/1606.05908.pdf*

[6] J. Duchi, Derivations For Linear Algebra and Optimization, (2018) Available at: *http://web.stanford.edu/~jduchi/projects/general_notes.pdf*

[7] J. C. Dunn, Well-Separated Clusters and Optimal Fuzzy Partitions, Journal of Cybernetics, 4 (1): pp. 95–104, ()1974).

[8] A. Géron, Hands-on Machine Learning with Scikit-Learn and Tensorflow, 2nd Ed. O'Reilly, (2017).

[9] N. Goix, How to Evaluate the Quality of Unsupervised Anomaly Detection Algorithms?, (2016), Available at: *https://arxiv.org/pdf/1607.01152.pdf*

[10] J. Goodfellow, J. P. Abadie, M. Mirza, B. Xu, D. W. Farley, S. Ozair, A. Courville, Y. Bengio, Stat. ML. arXiv:1406.2661, Generative Adverserial Nets, (2014).

[11] G. E. Hinton, P. Dayan, B. J. Frey and R. M. Neal, The wake-sleep algorithm for unsupervised neural networks, (1995), Available at: *http://science.sciencemag.org/content/268/5214/1158http://science.sciencemag.org/content/268/5214/1158http://science.sciencemag.org/content/268/5214/1158*

[12] S. Jianlin, Variational Self-Encoder: One-Step Clustering Scheme, (2018), Available at: *https://kexue.fm/archives/5887(Chinese)*

[13] Z. Jiang, Y. Zheng, H. Tan, B. Tang and H. Zhou, CS arXiv 1611.05148, Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering, (2016).

[14] P. Kingma, M. Welling, Stat. ML. arXiv:1312.6114, Auto-Encoding Variational Bayes, (2014).

[15] N. Laird, and D. Rubin, Maximum Likelihood from Incomplete Data via the EM Algorithm, (1977), Available at: *http://web.mit.edu/6.435/www/Dempster77.pdf*

[16] F. F. Li, J. Johnson and S. Yeoung, Lecture 13, Generative Models, (2017), Available at: *https://www.youtube.com/watch?v=5WoItGTWV54*

[17] F. T. Liu, K. M. Ting and Z.-H. Zhou, Isolation forest, In: Proc. of the 8th IEEE International Conference on Data Mining (ICDM'08), Pisa, Italy, pp.413-422 (2008).

[18] S. P. Loyd, Least squares quantization in PCM, (1982), Available at: *https://scinapse.io/papers/2150593711*

[19] W. McCulloch and W. Pitts, A Logical Calculus of the Ideas Immanent in Nervous Activity, Bulletin of Mathematical Biophysics, 5, pp. 115-133, (1943).

[20] L. Mart, N. Sanchez-Pi, J. M. Molina, and A. C. Bicharra Garcia. Anomaly detection based on sensor data in petroleum industry applications. Sensors, 15(2): 2774–2797, (2015).

[21] Q. Nguyen, K. Lim, D. Divakaran, K. Low and M. Chan, GEE: A Gradient-based Explainable Variational Autoencoder for Network Anomaly Detection, (2019), Available at: *https://arxiv.org/pdf/1903.06661.pdf*

[22] E. Norlander, *Clustering and Anomaly Detection on Financial Trading Data using the Conditional Latent Space Variational Autoencoder*, master thesis, Lund University, (2019).

[23] K. Pearson, On Lines and Planes of Closest Fit to Systems of Points in Space, The London Edinburgh Dublin Philosophical Magazine and J. of Sc., 2 (11), pp. 559-572, (1901).

[24] A. Rosenberg and J. Hirschberg, V-Measure: A conditional entropy-based external cluster evaluation measure, (2007), Available at: *https://www.aclweb.org/anthology/D07-1043*

[25] F. Rosenblatt, Perceptrion: A Probabilistic Model For Information Storage And Organization In The Brain, Psycological Review, 65(6), (1958).

[26] D. E. Rumelhard, G. E. Hinton and R. J. Williams, Learning representations by back-propagating errors, Nature, 323, pp. 533-536, (1986).

[27] B. Scholkopf, R. Williamson, A. Smola, J. Shawe-Taylor and J. Platt, Support Vector Method for Novelty Detection, (2000), Available at: *http://papers.nips.cc/paper/ 1723-support-vector-method-for-novelty-detection.pdf*

[28] E. Schubert, A. Zimek, and H.-P. Kriegel. Local outlier detection reconsidered: ageneralized view on locality with applications to spatial, video, and network outlier detection. Data Min. Knowl. Discov, 28: 190-237, (2014).

[29] R. Shu, Density Estimation: Variational Autoencoders, (2018), Available at: *http://ruishu. io/2018/03/14/vae/*

[30] R. Shu, Gaussian Mixture VAE: Lessons In Variational Inference, Generative Models and Deep Nets, (2016), Available at: *http://ruishu.io/2016/12/25/gmvae/*

[31] K. Sohn, X. Yan, H. Lee, Learning Structured Output Representation using Deep Conditional Generative Models (2015), Available at: *https://pdfs.semanticscholar.org/3f25/ e17eb717e5894e0404ea634451332f85d287.pdf*

[32] *A. Sopasakis, Traffic demand and longer-term forecasting from real-time observations, Sopasakis, A., Conf. on Time Series and Forecasting, Granada, Vol. 2. pp. 1247-1259 (2019)*

[33] *A. Sopasakis, M. A. Katsoulakis, Information metrics for improved traffic model fidelity through sensitivity analysis and data assimilation, Transportation Research Part B: Methodological 86, 1-18 (2016)*

[34] L. Tiao, A Tutorial on Variational Autoencoders with a Concise Keras Implementation, (2018), Available at: *https://tiao.io/post/ tutorial-on-variational-autoencoders-with-a-concise-keras-implementation/*

[35] D. Train, Variational auto-encoders do not train complex generative models, (2016), Available at: *http://dustintran.com/blog/ variational-auto-encoders-do-not-train-complex-generative-models*

[36] J. Xie, R. Girshick and A. Farhad, CS arXiv 1511.06335, Unsupervised Deep Embedding for Clustering Analysis, (2015).

[37] H. Zenati, C.-S. Foo, B. Lecouat, G. Manek, V. R. Chandrasekhar, Efficient GAN-Based anomaly detection Workshop track, ICLR (2018).