Extending a Variational Autoencoder with a Gaussian Mixture model to cluster Nightingale's songs

Gaetano Signorelli

Abstract. The structure of the latent space produced by a previously tested VAE (Variational Autoencoder) has showed a relevant pitfall for clustering data coming from nightingales' songs: by compressing the data through the encoder, samples from different clusters are distributed following the same Gaussian distribution, scarcely separated in the latent space, and so producing hugely unsatisfactory scores after a clustering algorithm is called. This work shows the attempts of improving the VAE architecture by using a Gaussian Mixture as prior, with the idea of producing a more suitable latent representation for the purpose. All the advantages and the scalability issues are discussed, as well as the new results.

1. Introduction to the problem

One of the architectures that have been adopted, trying to solve the clustering problem on nightingales' songs, revolved around a Variational Autoencoder (VAE) [1], in its vanilla form. Among all the approaches, this one has led to the worse results, also taking into account the general expectations.

The main problem that has emerged, and that negatively influenced the clustering scores, was the organization of the latent space. Especially, one of the greatest advantages of a VAE compared to a standard Autoencoder (AE) [2], is that the former organizes the latent space, following a Gaussian distribution that contrasts overfitting at the same time. Since keeping the latent space well structured is a fundamental point for a pre-clustering dimensionality reduction, VAE has been preferred to an AE, that tends to rearrange it conveniently (to reduce much more the reconstruction loss).
This choice, though, has brought another, very similar, issue: a Gaussian arrangement of the samples tends to place all the images in the dataset into a big single cluster (around the mean of the features), creating an even worse space for all the possible clustering techniques; and explaining the extremely low values for the silhouette score.

2. Results without regularization

The very first approach that has been tried, in order to improve results on an autoencoder-based architecture, is the training of the same VAE **removing the Kullback-Leibler** divergence regularization. Basically, this operation approximately transforms the model into a standard AE,

since, to further reduce the loss, the standard deviations of all the features tend to converge to 0.

This simple adjustment has considerably raised the results, overcoming those achieved with the *baseline*, but remaining much below those achieved with the main model (*NSCNet*).

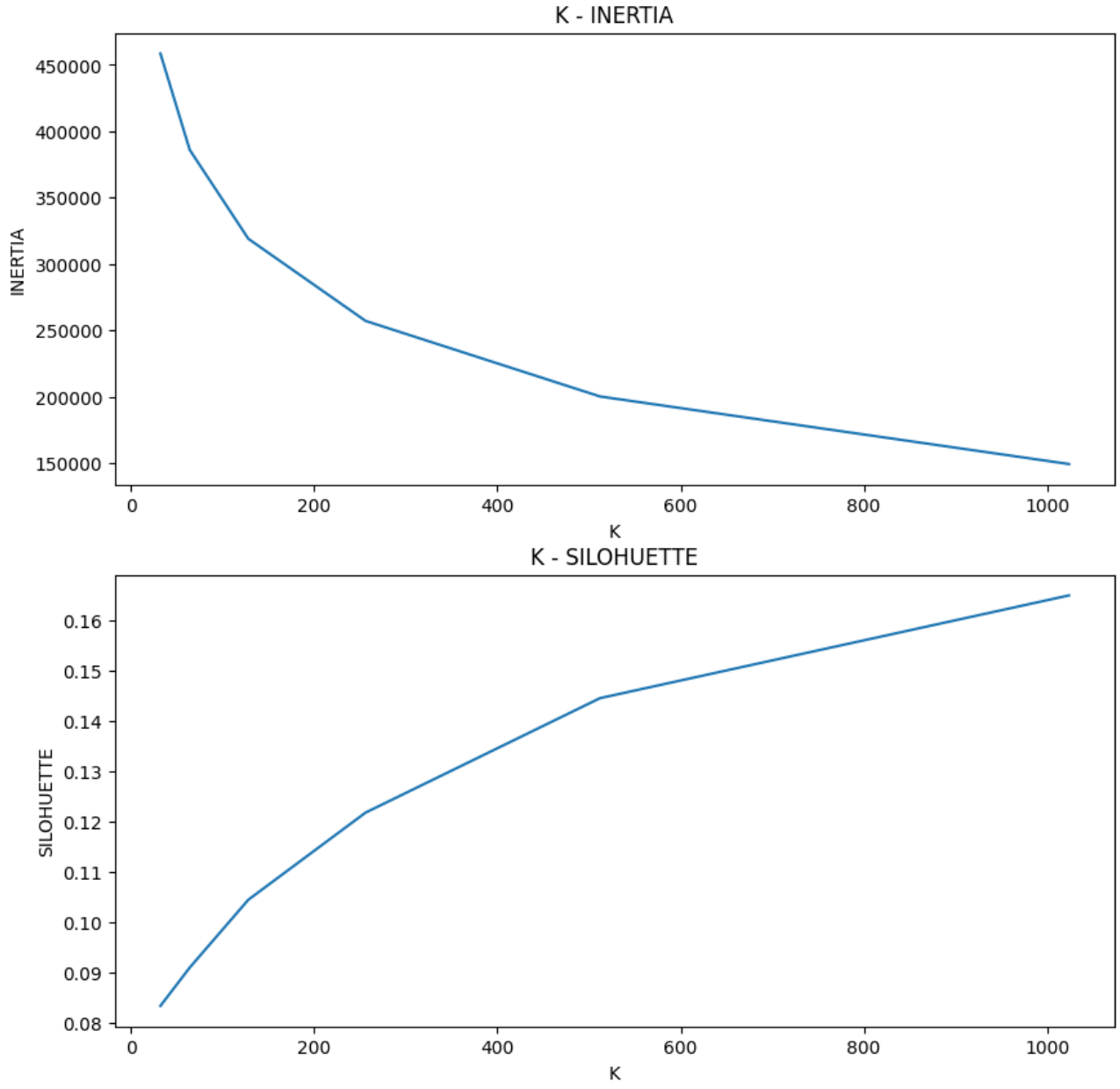

Figure 1 - K-means results removing the KL regularizer

As shown in *Figure 1*, the silhouette score arrives to a maximum of *0.16* for *1024* clusters (using the *K-means* technique, while the *DBSCAN* remains hugely unsatisfactory, with negative scores), with a small elbow for both silhouette and inertia on *512*. These results are slightly better than those obtained with the baseline but, although being also better than those coming from the previous VAE, stay below an acceptable threshold.

3. Variational Autoencoders with Gaussian Mixture Model

The main attempt to enhance the VAE has revolved around the implementation of a specific variation of it, so to have a much better compression of the data for the clustering step. To do so, a new architecture has been implemented, based on Variational Deep Embedding (***VaDE***), proposed in [3].

The general idea has emerged in recent years as a good exploitation of VAEs for Deep Clustering problems.

3.1 Theoretical approach

The main novelty introduced by [3] is the combination of VAE with a **Gaussian Mixture model** for clustering, generalizing the VAE with Mixture-of-Gaussians prior replacing the single Gaussian prior. The idea is to have a latent space where samples coming from different clusters are separated in different Gaussians, keeping an organization that is more suitable for a clustering task, that, in addition, can be accomplished by the same model, which, knowing in advance all the means and variances of all the Gaussians of the mixture, is able to determine, by looking at the compressed (encoded) representation of each sample, the probability of coming from each of them. It must also be noticed that, as for the classical VAE, each Gaussian is actually a multivariate one, since there are more dimensions (features) involved.

Thus, in this approach, the *Kullback-Leibler* divergence term of the loss function is computed with respect to a mixture of Gaussians, and this is achieved by using three more (learnable) parameters:

a. μ: this parameter is a matrix containing the **mean** of each Gaussian for each of the features;
b. σ: this is another matrix of the same type as the previous one, containing the **variances**;
c. π: this parameter is used in all the mixture models, and it represents the **weight** associated to each of the Gaussians (normalized so that the total sum is equal to *1*) and it can also be considered as the prior distribution about the clusters.

Calling z the features of an encoded sample, and c the Gaussian Mixture model, the architecture is able to compute:

$$p(z|c) = \frac{e^{-\frac{1}{2}\sum_{d}^{Dim}\frac{(z-\mu)^2}{e^{\sigma}}-\sigma}}{2\pi} \quad (1)$$

Then, the probability of belonging to each cluster (Gaussian) is given by γ:

$$\gamma = p(z|c)\pi \quad (2)$$

This makes the network capable of executing the clustering task (by taking the argmax from γ).

3.2 Results on MNIST

Once implemented, the new model has been tested on a sample dataset, the popular MNIST [4], trying to reproduce results from the original paper [3].

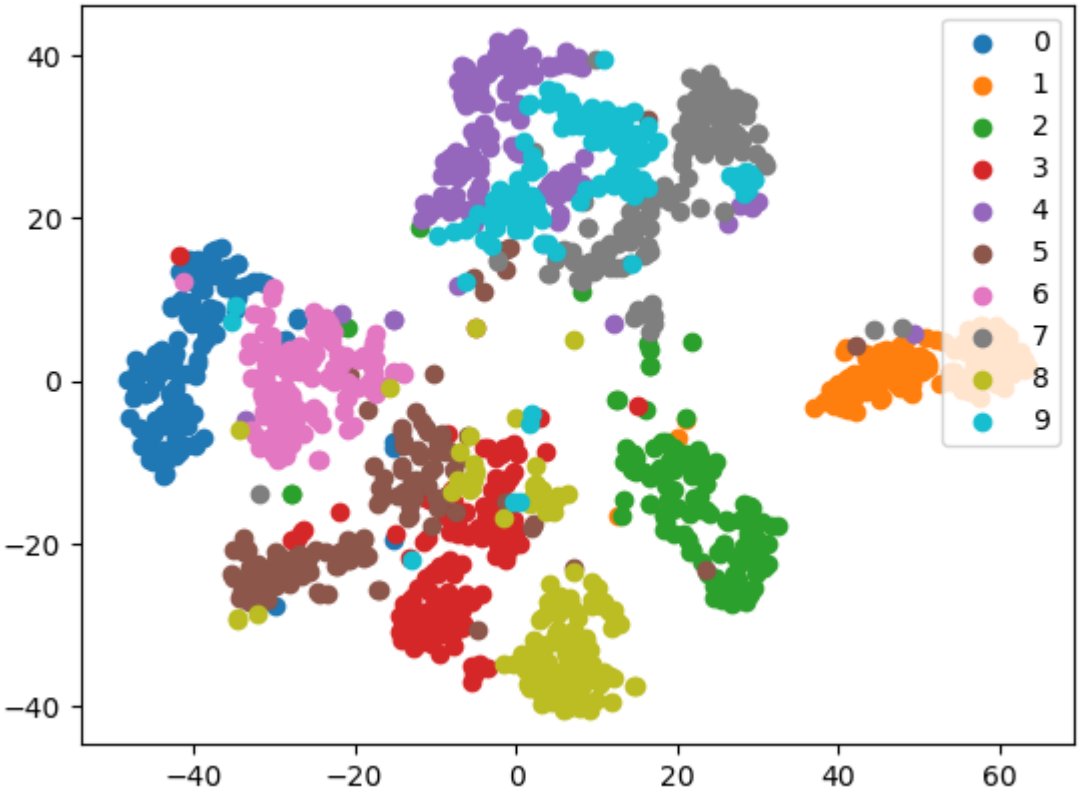

Figure 2 - Results of VaDE on MNIST

As it can be noticed in *Figure 2*, the samples have been almost perfectly separated into different clusters, showing the power of the architecture.

3.3 Advantages

The implemented approach has, at least from a theoretical point of view, a considerable number of advantages, compared to the other strategies:

a. By computing γ, the model can establish, based on its internal Gaussian mixture representation, the probability of a sample to belong to each cluster. In other words, it can be exploited as an **end-to-end model** to do both dimensionality reduction and clustering (as for the *NSCNet*), making the use of a secondary step (external clustering) optional.

b. Even if the number of clusters must be specified in advance, the prior probability associated to some of the clusters (some Gaussians) can be set to *0*, basically reducing the number of unique clusters.
c. This is the single model, among all of those that have been tried, to allow a **knowledge injection**. In particular, one of the parameters (π), represents the prior distribution of each cluster and it can be regularized or manipulated to fit some specific distribution. The most noticeable idea could be forcing it into a **Zip's law**, coherently with the hypothesis born from *NSCNet's* best results.

4. First problem: underflow

After the implementation and the first tests on other simple datasets, making use of *VaDe* to solve the clustering task on nightingales' songs has led to a couple of problems and, consequently, many failures.

The first encountered problem was due to the higher number of dimensions and clusters compared to the previous experiments. Going down into the details, the probability, for each of the features, to belong to each of the Gaussians (clusters) was computed, following the referenced paper [3], using the formula (1). Which basically computes the Gaussian probability for each cluster and along the axis of all the features.

However, this computation has been affected by a side effect: summing the values obtained for all the features, the next step involved the computation of e over a highly negative value, causing an underflow approximation that set all the probabilities to zero.

This issue has been solved by replacing the sum with a mean in (1):

$$p(z|c) = \frac{e^{-\frac{1}{2Dim}\sum_{d}^{Dim}\frac{(z-\mu)^2}{e^{\sigma}}-\sigma}}{2\pi} \quad (3)$$

By doing this, the maths behind the computation of conditioned probabilities has remained the same, since a higher mean along the features with respect to a specific Gaussian was still associated with a higher probability to belong to that Gaussian and vice versa, as it happened in (1).

5. Second problem: trivial parametrization

Differently from the simpler cases, with a higher number of clusters and a higher number of dimensions, the convergence to an optimal solution following closely the number of clusters given in input becomes very unstable.

The model finds a trivial path to minimize both the reconstruction loss and the *Kullback-Leibler* loss (with respect to the Gaussian mixture). It creates a complete unbalance toward one of the clusters by maximizing its prior probability, so that that cluster always takes the upper hand even after the computation in (2). In other words, the model tries to mimic what occurred in the vanilla VAE, also leading to a bad organized space and a single cluster as the output.

To tackle this issue, many attempts have been done:

- Since the given dataset may be highly skewed due to a supposed Zipf's law distribution, to mitigate the trivial parametrization coming from batches made up of a single cluster, a uniform cluster sampler has been implemented, based on *pseudo-labels* produced by the NSCNet. This preventative measure had no sensible improvements, making more probable the theory of the curse of dimensionality (see later).
- Other regularizers have been tried. Especially those based on the **entropy**, computed as expressed in (4) along the cluster-related axis: the entropies of the conditioned probabilities (probability of each set of features to belong to each Gaussian), the prior probabilities π and the likelihood probabilities γ have been forcefully manipulated to output a more reasonable final distribution of the clusters, instead of a single one dominating. Particularly, the entropy over the conditioned probabilities (along the clusters' axis) has been decreased, as well as for γ, adding a term to the loss function, trying to avoid a uniform distribution (meaning that each sample has the same probability to belong to each of the Gaussians), which has also been encountered in some experiments; while the entropy of π has been slightly increased (with a smaller loss term), since, as already pointed, this distribution tent to be too unbalanced in favour of a single cluster (very low entropy). All these regularizers have been tried in different combinations and with different weights parameters. The overall results slightly improved, with an increasing in the number of identified clusters, which still stayed very low, especially by increasing the expected number of clusters in input. Generally, the trivial parametrization has been only slightly reduced with this technique, with the model still oriented toward a simple solution with few clusters.
- A last regularizer has been tested with no success, trying to increase the number of unique clusters in the entire dataset. Unfortunately, it did not work since a *"unique"* mathematical operation was needed, but it is not differentiable and so it cannot be a part of the loss function.

$$H(X) = -\sum_{i=1}^{n} P(x_i)\log\left(P(x_i)\right) \quad (4)$$

6. Failures interpretation

After many tests, the more reasonable hypothesis is that by increasing the number of dimensions, the model becomes affected by the **curse of dimensionality**: with many features (the effect starts being strong with about *32* features), the distance between a compressed sample and all the Gaussians of the mixture is very similar. This should be expected with a much higher number of dimensions, but this is probably due to the complexity of the problem, which is also highly influenced by the number of clusters, that forces the model into the trivial parametrization.

The higher the dimensions and the clusters number, the higher the complexity (which seems to grow exponentially) and so the choice of the most trivial solution.

This hypothesis has also been tested practically, by reducing drastically both the number of features and clusters (separately) and getting a much better silhouette score, even though results were still not satisfying, because of the poor compression (too few dimensions to distinguish and reconstruct the input images) or an unlikely number of clusters.

All these tests point toward the fact that the architecture proposed in [3] suffers from huge issues in scalability, being impractical, even if it has many theoretical advantages.

7. Other attempts

To solve the discussed issues and exploit some of the listed advantages, other approaches have been attempted.

7.1 Knowledge injection

One of the main positive points about the use of *VaDE* as clustering architecture is the possibility to force a certain prior known distribution of the clusters, doing a knowledge injection as already discussed.

Hence, one of the attempts revolved around using the **Zipf's law** (5) as a prior for π:

$$f(k; s; N) = \frac{1}{k^s \sum_{n=1}^{N} (\frac{1}{n^s})} \quad (5)$$

With this distribution, the weights of π have been replaced by a single parameter s, controlling the Zipfian, that is posed as a constraint for the prior distribution.

It must be noticed that, even though this approach did not solve the problems of the model, it represents an interesting point in case of a similar and more scalable (in terms of dimensions and number of clusters) neural network.

7.2 Separated clustering

Having a defective end-to-end architecture, an alternative test on it has been executed by using it just to compress the data and then, in the same way as done for the baseline and the vanilla VAE, using a clustering algorithm to cluster and evaluate results, separately. Which, as expected, were close to those coming from the VAE, because of the explained tendency of collapsing all the Gaussians of the mixture into a single one.

7.3 Parameters initialization and pre-training

An ultimate test involved the initialization of the trainable Gaussian parameters π, σ and μ by applying the *PCA* to the original dataset, calling the Gaussian Mixture clustering over it and finally extracting those parameters. Moreover, they have not been trained during the first epochs, dedicated to pre-training only on input reconstruction.

These methods have also been suggested by [3], but their contribution has been very low.

8. Results and conclusions

After all the afore-mentioned tests, the results were quite unsatisfactory and similar to those of the VAE, as it can be observed in *Figure 3*:

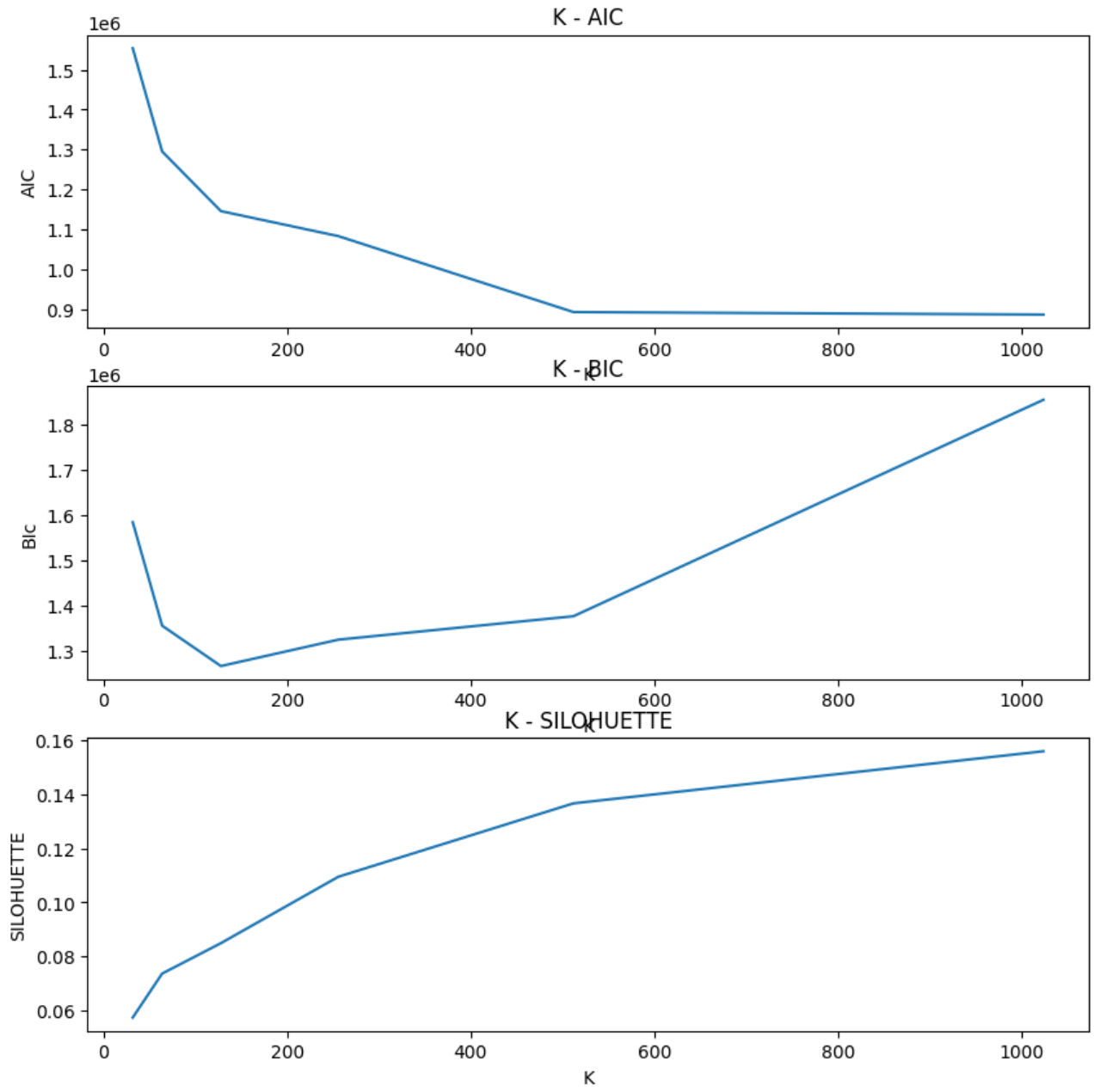

Figure 3 - Best results using VaDE

These results have been produced by compressing the data with the encoder of the *VaDE* and using an external Gaussian Mixture model clustering algorithm (the internal one recognized only a limited number of clusters due to the discussed trivial parametrization).

In conclusions, it has emerged a big weakness in the scalability of the proposed *VaDE* model, suffering in case of a relative high number of dimensions and clusters. On the other hand, this work has highlighted some interesting points that may become the key for an improvement in the future, used in conjunction with a more stable and performing architecture of the same kind. Those key points are:

- The possibility of injecting knowledge through access on the prior distribution of the clusters;
- The flexibility in the number of clusters returned by the model (that can be different from the one in input);
- The strength of a model that does both data compression and clustering, in an end-to-end approach.

References

1. Kingma, et al.: Auto-Encoding Variational Bayes (2014)
2. Bank, et al.: Autoencoders (2003)
3. Jiang, et al.: Variational Deep Embedding: An Unsupervised and Generative Approach to Clustering (2017)
4. LeCun, et al.: The MNIST database of handwritten digits (1998)