CAPTURING LABEL CHARACTERISTICS IN VAEs [2]

Mohamed Benyahia-Tancrede Martinez-Hadrien Levechin

December 2024

1 Introduction

1.1 Objective of the research paper

Variational Autoencoders (VAEs) have become a quite popular tool for generative modeling. Despite their success, the basic architecture of VAEs does not provide a tool to generate data in a supervised way, having labels as supplementary information to learn from.
A common approach for conditional data generation is to add a projection of the one-hot labels $OHE(y)$ through an MLP to the latent variable z ,giving the encoder $z + MLP(OHE(y))$.
However, while this method shows promising results in some cases, one may question whether using label information only after generating the latent variable fully captures the additional knowledge provided. Also, how can we ensure that images with similar latents but slight differences in labels remain "close" after generation?
To answer this problematic , the authors of the paper [2] describe a novel architecture based on VAEs : The characteristic capturing VAE (CCVAEs), allowing to learn a latent representation containing both a characteristic latent, involving information about the labels and a style latent, representing general information for smooth generation.
Additionally, to have a more general model that relies not only on supervised-data, the authors propose an approach to learning in a semi-supervised way, allowing to learn on a broader range of data sample.

1.2 Goal of our work

In this work, we propose an simpler implementation of the (CCVAEs) using inherent knowledge about our dataset. We also evaluate the influence of β-regularization to performance of the model.Mohamed wrote the background, the results and discussion parts, implemented and tested the first version of model with the non-split latent space. Tancrede made the division of the latent space, tested the adapted model, compared the beta influence with the results

of a non-separated latent, and wrote the model part in the report. Hadrien wrote the experiments and implementation part and worked on the graph in the poster, optimized and improved the model after running a lot of tests with different values for the hyperparameters.

2 Background and Related Work

2.1 Background :

Variational Autoencoders combine deep autoencoders with generative latent-variable models to learn data representations as distributions.They model the joint distribution of data $p(x, z) = p(z)p(x|z)$ using :
Encoder : Approximates the intractable posterior $p(z|x)$.
Decoder : Parametrizes the likelihood $p(x|z)$ using a neural network. The objective is to maximize the marginal likelihood via the Evidence Lower Bound (ELBO), which balances reconstruction and regularization.

2.2 Existing work :

Semi-Supervised VAEs (SSVAEs): Approaches like Kingma and al. (2014) [3] and Siddharth and al. (2017) [5] treat labels as latent variables and use them to improve classification and data generation.
Hierarchical VAEs: Sønderby and al. (2016) [6] introduced hierarchical latent spaces to improve flexibility.
Manipulation in SSVAEs: Prior methods use the decoder for generating data after label intervention, but they often fail to disentangle meaningful characteristics.
Multimodal and Auxiliary Supervision: Wu and Goodman (2018) [7] introduced MVAE which handled multimodal data but lacked explicit classifier-aligned latent structures.
Domain-Specific and GAN-Based Models: Ilse et al. (2019) [1] introduced DIVA which focused on domain-generalized classifiers but required additional regularization. Also, GAN-based approaches structured latents but lacked capabilities like classification or conditional generation.

3 Proposed method

3.1 Model definition

The main goal of the paper is to propose a modified VAE's architecture, the CCVAE, that allows to have an interesting representation of given label into the latent space.
We denote x the input of the network, z the latent variable, y the label associated to the input.

We want to divide our latent into $z = \{z_c, z_{\setminus c}\}$ where z_c represents the characteristic latent, e.g corresponding to label characteristics representation in latent space.
As usual with VAE's, we want to learn $q_\phi(z|x)$ for the encoder and $p_\theta(x|z)$ for the decoder. Moreover, we want to learn the characteristic latent as $p_\psi(z_c|y)$ as well as the prior $p(y)$. We denote the posterior latent distribution $q_\gamma(y|z_c)$.
Also, we partition latent space to disentangle label such that :

$$p(z_c|y) = \prod_i p(z_c^i|y^i) \quad (1)$$

3.2 Loss computation

Similarily to VAE's ELBO, we can find that for supervised case :

$$logp_{\theta,\psi}(x,y) \geq \mathbb{E}_{q_\phi(z|x)}[log(\frac{p_\theta(x|z)p_\psi(z|y)p(y)}{q_{\gamma,\phi}(z|x,y)})] = \mathcal{L}_{\text{CCVAE}}(x,y) \quad (2)$$

and for unsupervised case :

$$logp_\theta(x) \geq \mathbb{E}_{q_\phi(z|x)q_\gamma(y|z_c)}[log(\frac{p_\theta(x|z)p_\psi(z|y)p(y)}{q_{\gamma,\phi}(z|x,y)})] = \mathcal{L}_{\text{CCVAE}}(x) \quad (3)$$

So that our total loss is :

$$\mathcal{L}(\mathcal{S},\mathcal{U}) = \sum_{(x,y)\in\mathcal{S}} \mathcal{L}_{\text{CCVAE}}(x,y) + \sum_{x\in\mathcal{U}} \mathcal{L}_{\text{CCVAE}}(x)$$

where $\mathcal{S}$ is the set of supervised training data and $\mathcal{U}$ the unsupervised one.

3.3 Model simplification

As for our testing, we propose a simplified model, working only on one label at a time, without the need to disentangle the label space as in (1).
For the unsupervised case, since we worked with dataset containing only few labels, we have worked with a sharper bound than (3) using the standard ELBO bound (K is the number of classes):

$$\mathcal{L}(x) = -\mathbb{E}_{p(z)}(logp_\theta(x|z)) + \mathcal{D}_{KL}(q_\phi(z|x)|p(z)) + \mathbb{E}_{q_\phi(z|x)}\left[\mathrm{H}\left(q_\gamma(y|z_c)\right)\right]$$

with $p(z) = p(z_{\setminus c}) \sum_y p(z_c|y)p(y)$
Furthermore, we assume that y is uniformly distributed among labeled data.
We then add a β regularization, to have our final unsupervised data loss :

$$\mathcal{L}(x) = -\mathbb{E}_{p(z)}(logp_\theta(x|z)) + \beta\mathcal{D}_{KL}(q_\phi(z|x)|p(z)) + \mathbb{E}_{q_\phi(z|x)}\left[\mathrm{H}\left(q_\gamma(y|z_c)\right)\right]$$

For the supervised case, we add the cross entropy loss to enforce label characterization :
$\mathcal{L}(x,y) = -\mathbb{E}(logp_\theta(x|z)) + \beta\mathcal{D}_{KL}(q_\phi(z_{\setminus c}|x)|p(z_{\setminus c}))$
$+\beta\mathcal{D}_{KL}(q_\phi(z_c|x)|p(z_c|y)) + CE(y, decoder(z))$

4 Experiments and Implementation

In this section, we describe the network implementation, the training procedure and the data set used. The aim is to propose an implementation for training the CCVAE described in the article and in the previous sections.

4.1 Data Base

We chose to train the model on the MNIST dataset [4]. The scenario is as follows: a 10-digit classification. The labels provided are the number associated with the image. The ground truth is used as the label.

We have divided the training set into 2.

- Supervised subset (S). A portion of the training data where each sample (x,y) is labeled.
- Unsupervised subset (U). The remaining data is unlabeled, encouraging the model not to overlearn on the supervised data, and to build a structured latent space.
- Finally, we also retain a portion for a sample validation set.

4.2 Model architecture and implementation details

The **CCVAE** model consists of four main modules:

- **Encoder :** It maps the input images $x$to a latent representation z, divided into z_c (label-related component) et $z_{\setminus c}$. The encoder uses a series of convolution layers followed by linear layers to produce the parameters (μ_q, σ_q) of the posterior approximation.
- **Decoder :** It reconstructs images from the latent vector z.The decoder projects z into an intermediate space before using transposed convolution layers to generate the reconstructed images.
- **Classifier :** A linear layer uses z_c to predict labels in supervised data. This ensures that z_c captures discriminating information for the labels.
- **Conditional prior (CondPrior) :** This module generates the parameters (μ_p, σ_p) of the conditional prior $p_\psi(z_c|y)$, from a label embedding layer.

All components are implemented in **PyTorch** with `ReLU` activations and convolution and linear layers.

4.3 Training procedure

The training follows a semi-supervised scheme, where we work on one label at a time without requiring label space de-interlacing. We use the standard ELBO bound for unlabeled data and add a regularization β to balance reconstruction and KL regularization.

The overall loss is written as follows:

- **For unsupervised data :**

$$\mathcal{L}(x) = -\mathbb{E}p(z)\left[\log p\theta(x|z)\right] + \beta\,\mathcal{D}KL\left(q\phi(z|x)\|p(z)\right) + \mathbb{E}q\phi(z|x)\left[\mathrm{H}(q_\gamma(y|z_c))\right]$$

- **For supervised data :**
$\mathcal{L}(x,y) = -\mathbb{E}(logp_\theta(x|z)) + \beta\mathcal{D}_{KL}(q_\phi(z_{\setminus c}|x)|p(z_{\setminus c}))$
$+\beta\mathcal{D}_{KL}(q_\phi(z_c|x)|p(z_c|y)) + CE(y, decoder(z))$

5 Results and Discussion

During our experiments, we were interested in studying the impact of two hyperparameters : the β value in the KL divergence term and the size of the latent space which is equal to 2*latent dim.
On one hand, a smaller value of β, closer to 0, will make the VAE behave more like a standard autoencoder, focusing on reconstruction accuracy. This means the latent space might not adhere closely to the desired prior distribution $p(z|y)$, affecting the model's ability to encode and generate images according to specific labels.In the opposite, a larger value will enforce more regularization, leading to potentially more distortion and noise in the generated images.

Figure 1: Reconstructed images $\beta = 0.05$

Figure 2: Reconstructed images $\beta = 0.4$

Figure 3: Conditionally generated images $\beta = 0.001$

Figure 4: Conditionally generated images $\beta = 0.05$

By observing the output images, we can say that lowering β has led to sharper and more defined digit images, indicating better reconstruction quality. There is less distortion and less blurriness.But if we lower it too much, $\beta = 0.001$, we suffer from poor label conditioning. Also, the transitions between the labels 0 and 1 are less smooth and more abrupt. If we increase β , the images seem

Figure 5: Conditionally generated images $\beta = 0.2$

Figure 6: Conditionally generated images $\beta = 0.4$

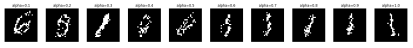

Figure 7: Interpolated images $\beta = 0.05$

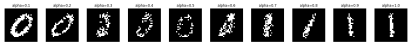

Figure 8: Interpolated images $\beta = 0.4$

to change gradually from one label to another, indicating a smoother and well-regularized latent space; as you can see below in the interpolated images. However, lowering β means the model can focus more on capturing features accurately so that's why we achieved high classification accuracy earlier than with high values of β.

On the other hand ,a higher latent dimension helps to increase the capacity to capture complex and detailed features of the input data. But, it can also mean more noise and overfitting (See Appendix). That's what we observe in the images in the appendix.

Remark : (See Appendix) The validation loss is lower than the train loss; because here in our case, the training is done in a semi-supervised setting; but the validation is done only in supervised setting.

6 Conclusion

We reimplemented a simplified version of the CCVAE model and evaluated its performance on the relatively simple MNIST dataset. The results were encouraging, demonstrating the model's ability to capture label characteristics effectively in a controlled setting. However, several challenges remain. The model's reliance on high-quality labels poses a limitation, as noisy or inconsistent labels could degrade performance. Additionally, scaling the approach to more complex datasets with intricate or high-dimensional labels may require further refinement to address potential computational and modeling challenges.

7 Appendix :

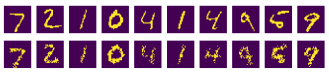

Figure 9: Reconstructed images latent dim = 16

Figure 10: Reconstructed images latent dim = 128

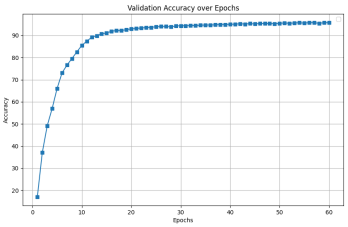

Figure 11: Validation accuracy $\beta = 0.05$

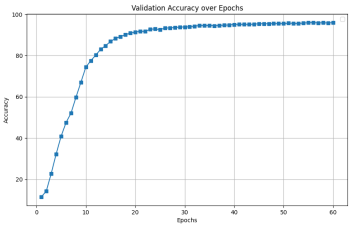

Figure 12: Validation accuracy $\beta = 0.4$

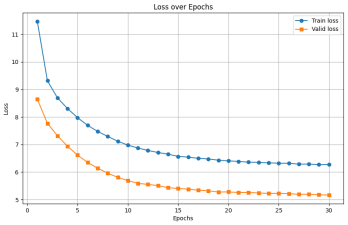

Figure 13: Train and Validation losses latent dim = 16

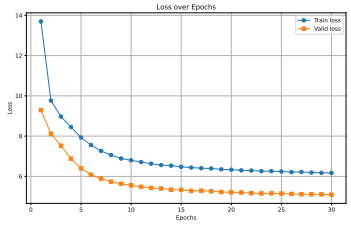

Figure 14: Train and Validation loss latent dim = 128

References

[1] Micaela Ilse, Piyush Kothari, Ruben Houthooft, and Alessandro Sordoni. Diva: Domain-invariant variational autoencoders. In *Proceedings of the 36th International Conference on Machine Learning (ICML)*, 2019.

[2] Tom Joy, Sebastian Schmon, Philip Torr, Siddharth N, and Tom Rainforth. Capturing label characteristics in {vae}s. In *International Conference on Learning Representations*, 2021.

[3] D.P. Kingma, D.P. Welling, I. Rezende, S. Mohamed, and M. Iyyer. Semi-supervised learning with deep generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 27, 2014.

[4] Yann LeCun, Corinna Cortes, and Christopher JC Burges. The MNIST database of handwritten digits. In *The MNIST database*, 1998.

[5] N. Siddharth, D.P. Kingma, D.P. Rezende, S. Mohamed, M. Iyyer, R. Salakhutdinov, and M. Welling. Learning disentangled representations in vaes with arbitrary latent variable dependencies. In *International Conference on Learning Representations (ICLR)*, 2017.

[6] S. K. Sønderby, L. Maaløe, A. Sørensen, M. Hoffman, and O. Winther. Ladder variational autoencoders. In *Neural Information Processing Systems (NeurIPS)*, 2016.

[7] Lin Wu and Noah D. Goodman. Multimodal variational autoencoders. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems (NeurIPS)*, 2018.