“Linear” Variational Autoencoder: Why One Should And Should Not Use Linear Models As Generative Model

Aukkawut Ammartayakun

Department of Data Science, Worcester Polytechnic Institute
aammartayakun@wpi.edu

Outline

1. Why do we care?
2. Autoencoder as a Pair of Parameterized Functions
3. Linear Model as Parameterized Functions
4. Linear Generative Model and Inference Results
5. Conclusion and Discussion

Why do we care?

Why do we care?

- Linear model is fast and easy to compute on CPU.

Why do we care?

- Linear model is fast and easy to compute on CPU.
- Deep learning is not that easy to compute on CPU.

Why do we care?

- Linear model is fast and easy to compute on CPU.
- Deep learning is not that easy to compute on CPU.
- If we can make the linear model to have really close performance model to the deep learning in a generative task, it should be good.

Autoencoder

Autoencoder

Autoencoder

Let say that we have a dataset which is $\mathbb{R}^n \supseteq \mathcal{D} = \{\mathbf{x}_1, \ldots, \mathbf{x}_m\}$. Define the parameterized function that captures the low-dimensional representation $\phi^e_{\boldsymbol{\theta}_1} : \mathbb{R}^n \to \mathbb{R}^p$ where $p \neq n$ and $\boldsymbol{\theta}_1$ as parameters. The autoencoder with parameter $\boldsymbol{\Theta} = \begin{bmatrix} \boldsymbol{\theta}_1 & \boldsymbol{\theta}_2 \end{bmatrix}^\top$ is the combination (or composition) of the encoding parameterized function $\phi^e_{\boldsymbol{\theta}_1}$ and the decoding parameterized function $\phi^d_{\boldsymbol{\theta}_2} : \mathbb{R}^p \to \mathbb{R}^n$ specifically,

$$\mathrm{AE}(\mathbf{x}) = \phi^d_{\boldsymbol{\theta}_2} \circ \phi^e_{\boldsymbol{\theta}_1}(\mathbf{x})$$

In practice, those said parameterized functions are generally the neural network model as the method of generating embedding and recovering the data is complex.

Linear Model as Parameterized Functions

Problem Statement

Given the previous definition, if one were to use *k*-rank matrix approximation for the data (i.e., *k*-PCA) for encoder

$$\mathbf{Z} = \hat{\phi}^{e}_{\boldsymbol{\theta}_1}(\mathbf{X}) = \sum_{j=1}^{k} d_j \mathbf{u}_j \mathbf{v}_j \approx \tilde{\mathbf{X}}$$

and use multiple multivariate linear regression,

$$\hat{\phi}^{d}_{\boldsymbol{\theta}_2}(\mathbf{X}) = \mathbf{Z}\boldsymbol{\theta}_2$$

This should still follow our definition of the autoencoder. Why does no one use this?

Empirical Initial Result

- Initial data: MNIST data, $\mathbf{x} \in \mathbb{R}^{28\times 28} \to \mathbb{R}^{784}$
  - Use PCA to approximate k-dimensional representation vector.
  - Multivariate multiple linear regression is then used to recover the data.

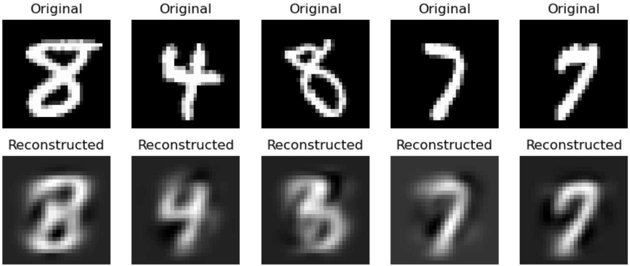

Figure 1: Naive approach

Latent Space Interpolation

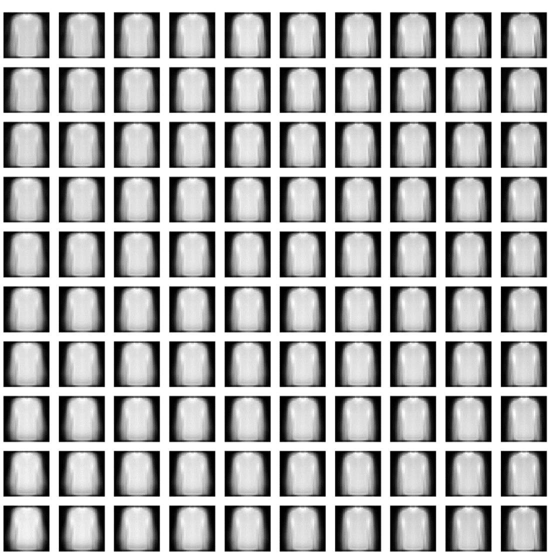

Figure 2: Latent space interpolation of the PCA and least square MMLR

Mathematical Interpretation of Initial Result

Postulate 1

Autoencoder where the encoder is PCA and the decoder is MMLR results in the reconstruction which lies in the span of the PC score.

Mathematical Interpretation of Initial Result

Postulate 1

Autoencoder where the encoder is PCA and the decoder is MMLR results in the reconstruction which lies in the span of the PC score.

Proof Let $\mathbf{X} \in \mathbb{R}^{n \times p}$ be a data matrix. k-rank approximation of $\mathbf{X}$ through PCA is

$$\tilde{\mathbf{X}} = \sum_{j=1}^{k} d_j \mathbf{u}_j \mathbf{v}_j = \mathbf{U}_k \mathbf{D}_k \mathbf{V}_k^\top$$

Least square estimate for $\mathbf{B}$ of the linear regression model $\mathbf{X} = \tilde{\mathbf{X}}\mathbf{B}$ is $\mathbf{B} = \left(\tilde{\mathbf{X}}^\top \tilde{\mathbf{X}}\right)^{-1} \tilde{\mathbf{X}}^\top \mathbf{X}$. Substitute in the $\mathbf{U}_k \mathbf{D}_k \mathbf{V}_k^\top$ to get

$$\begin{aligned}
\mathbf{B} &= \left((\mathbf{U}_k \mathbf{D}_k \mathbf{V}_k^\top)^\top \mathbf{U}_k \mathbf{D}_k \mathbf{V}_k^\top\right)^{-1} (\mathbf{U}_k \mathbf{D}_k \mathbf{V}_k^\top)^\top \mathbf{X} \\
&= \left(\mathbf{V}_k \mathbf{D}_k \mathbf{U}_k^\top \mathbf{U}_k \mathbf{D}_k \mathbf{V}_k^\top\right)^{-1} \mathbf{V}_k \mathbf{D}_k \mathbf{U}_k^\top \mathbf{X} \\
&= \mathbf{V}_k \mathbf{D}_k^{-1} \mathbf{U}_k^\top \mathbf{X}
\end{aligned}$$

The reconstruction then be $\tilde{\mathbf{X}}\mathbf{B} = \mathbf{U}_k \mathbf{D}_k \mathbf{V}_k^\top \mathbf{V}_k \mathbf{D}_k^{-1} \mathbf{U}_k^\top \mathbf{X} = \mathbf{U}_k \mathbf{U}_k^\top \mathbf{X}$ □

Linear Generative Model and Inference Results

Linear Generative Model

$$Y_{\mathsf{obs}} | \mathbf{X}_{\mathsf{M}}, \ell \sim \int \mathcal{N}\left(\mathbf{X}_{\mathsf{M}} \mathbf{B}_{\ell}^{\top}, \Sigma\right) q(\mathbf{B}_{\ell}) d\mathbf{B}_{\ell}$$

- In this study, we define the transformed data $\mathbf{X}_{\mathsf{M}}$ by model M to be as followed:
  - Principal Component Analysis (PCA)
  - Linear Discriminant Analysis (LDA)
  - Probabilistic Principal Component Analysis (PPCA)
- In this regression model, parameter **B** is treated as *random*.
  - The realization of those parameters are from the average of the sample drawn from the posterior distribution $q(\mathbf{B})$ by variational inference method.

Variational Inference (VI)

- The idea is to find the easier-to-calculate distribution $q(\theta)$ to approximate the true posterior distribution $p(\theta|x)$.
- This can be done by minimizing the KL divergence

$$\text{KL}(q(\theta)||p(\theta|x)) = \int q(\theta) \ln \left(\frac{q(\theta)}{p(\theta|x)} \right) \, d\theta$$

- $p(\theta|x)$ is intractable (well, we want to approximate it in the first place). However, using Bayes' rule on the KL divergence definition,

$$\begin{aligned} \text{KL}(q(\theta)||p(\theta|x)) &= \int q(\theta) \ln \left(\frac{q(\theta)p(x)}{p(x|\theta)p(\theta)} \right) \, d\theta \\ \text{KL}(q(\theta)||p(\theta|x)) &= \ln(p(x)) + \int q(\theta) \ln(q(\theta)) \, d\theta \\ &\quad - \int q(\theta) \ln(p(x|\theta)) \, d\theta - \int q(\theta) \ln(p(\theta)) \, d\theta \end{aligned}$$

Variational Inference (VI)

$$\begin{aligned}\mathrm{KL}(q(\theta)||p(\theta|x)) &= \ln(p(x)) + \int q(\theta)\ln(q(\theta))\,d\theta \\ &\quad - \int q(\theta)\ln(p(x|\theta))\,d\theta - \int q(\theta)\ln(p(\theta))\,d\theta \\ \ln(p(x)) &= \int q(\theta)\ln(p(x|\theta))\,d\theta + \int q(\theta)\ln(p(\theta))\,d\theta \\ &\quad - \int q(\theta)\ln(q(\theta))\,d\theta + \underbrace{\mathrm{KL}(q(\theta)||p(\theta|x))}_{\geq 0}\end{aligned}$$

- We call the lower bound of this log evidence *Evidence Lower BOund (ELBO)* which is what we want to maximize in order to minimize KL divergence.
- This can be done using an optimization algorithm like the Newton-Raphson or gradient-based method.

Latent Space Interpolation

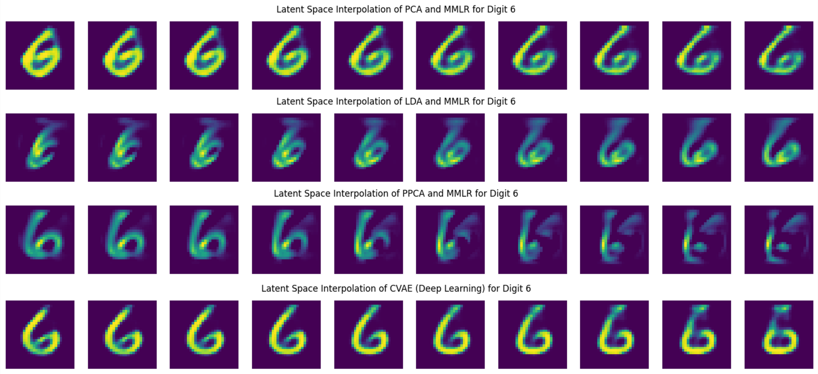
Latent Space Interpolation of PCA and MMLR for Digit 6
Latent Space Interpolation of LDA and MMLR for Digit 6
Latent Space Interpolation of PPCA and MMLR for Digit 6
Latent Space Interpolation of CVAE (Deep Learning) for Digit 6

Figure 3: Latent space interpolation of the digit 6 from different method

- They can generate new images within the latent space which is (potentially) unseen.
- Comparable result with deep learning-based approach.

Latent Space Interpolation (Con't)

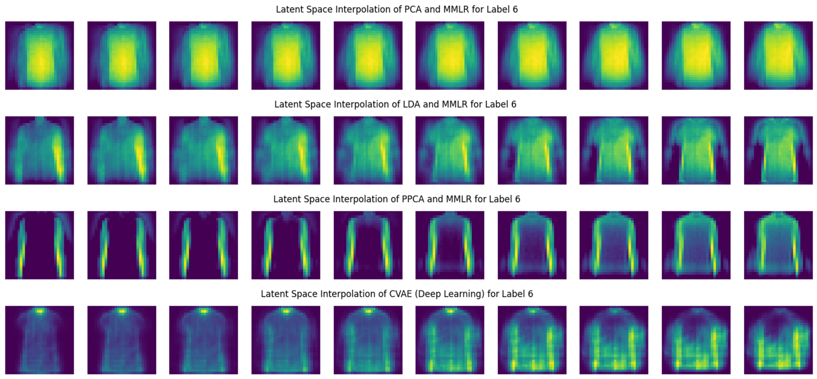
Latent Space Interpolation of PCA and MMLR for Label 6
Latent Space Interpolation of LDA and MMLR for Label 6
Latent Space Interpolation of PPCA and MMLR for Label 6
Latent Space Interpolation of CVAE (Deep Learning) for Label 6

Figure 4: Latent space interpolation of the label 6 from different method

- Here, the data are more complex. However, all the linear models still be able to produce the recognizable image.

Numerical Results

p-Kantorovich-Rubinstein metric (or Wasserstein distance) is used for the purpose of assessing the generative model

$$d_{F,p}(\mu,\nu) := \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{\mathbb{R}^n \times \mathbb{R}^n} \|x - y\|_p \; d\gamma(x,y) \right)^{\frac{1}{p}}$$

Here, we essentially find the minimum expectation of L^p-distance over a set of all coupling probability measures μ and ν.

For our purposes, $p = 1$ should suffice. Intuitively, from the optimal transport point of view, this is like saying how much of the *cost* it takes to transform the shape of the distribution from the measure μ to ν.

Numerical Result

Table 1: Numerical evaluation of the combination of linear models on the 1-Wasserstein distance (realism) and number of principal components on covariance for 95% total variation explained (variability) on handwritten digit MNIST dataset based on empirical distribution from 2000 samples of the prediction distribution on MMLR

Model	1-Wasserstein (realism)	Number of PCs on covariance for 95% total variation explained (variability)
PCA + Least squared MMLR	35.1612	9
PCA + VI MMLR	**24.4795**	10
LDA + VI MMLR	35.1802	5
PPCA + VI MMLR	35.2660	10
CVAE	35.1242	**13**

Conclusion and Discussion

Conclusion

- We can generate a lossy reconstruction of the initial data.
- However, we have shown that the viability of the generated images is restricted in the submanifold from the span of the PC score.
    - We also have shown, from numerical results, that adding stochastic to this "linear model" does move us out of that space.
- Here, while we have shown numerically that when $k \to n$, we would have lossless data. However, the trend is not uniform.
- Yet, this alone is enough to be competable with a deep learning model on the smaller-dimensional data while having fewer parameters to estimate.

Unexpected Result

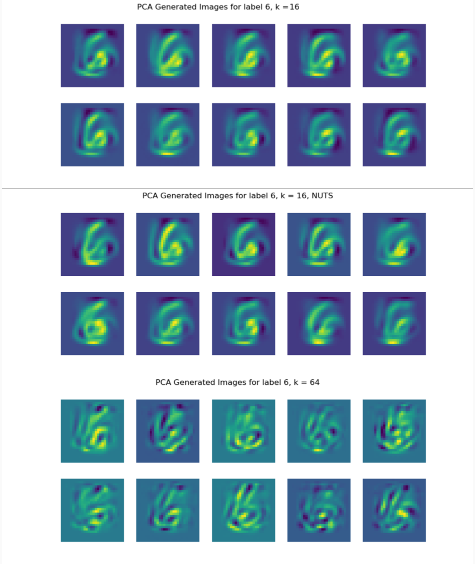
PCA Generated Images for label 6, k =16
PCA Generated Images for label 6, k = 16, NUTS
PCA Generated Images for label 6, k = 64

Figure 5: Inference result for k = 16 and 64

Discussion

- More studies and experiments are needed to actually show whether a linear model is enough or not. But one thing that can say (with some caution) is that stochastic element does prevent the collapsing of composition of linear mapping into one linear mapping.
- What if we try to project to the higher space first? Using Cover's theorem, arguably there should be a linear manifold that captures the distinctiveness of embedding. $\longrightarrow$ **Kernel method** (kind of)
- Also, our derivation of ELBO is from the KL divergence, since our objective is to minimize the expected L^p-distance on the two measures, why don't we use that as our objective for the optimization? $\longrightarrow$ **Wasserstein GAN** (kind of)

References i

Cottrell, G., Munro, P., and Zipser, D. (1987). Learning internal representations from gray-scale images: An example of extensional programming. *Proceedings of the Ninth Annual Cognitive Science Society Conference*, pages 461–473.

David M. Blei, A. K. and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877.

LeCun, Y. and Cortes, C. (1998). The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*.

Lucas, J., Tucker, G., Grosse, R. B., and Norouzi, M. (2019). Don't blame the elbo! a linear vae perspective on posterior collapse. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Sohn, K., Lee, H., and Yan, X. (2015). Learning structured output representation using deep conditional generative models. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc.

Wainwright, M. J. (2019). *Concentration of measure*, page 58–97. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press.