PCA-like Autoencoding

Ziyi Liu
Department of Computer Science and Technology
University of Cambridge
zl413@cam.ac.uk

Abstract

Principal Component Analysis(PCA) is a widely popular approach for dimensionality reduction by producing a set of orthogonal latent variables with ordered importance. Autoencoders (AE), which can also be used to learn a predetermined dimension of latent representation, are more capable of capturing complex non-linear features due to the flexibility of its encoder-decoder architecture. However, importance ordering of latent variables and independence are lost in simple autoencoders, and as such the main objective of this paper is to retain these two features to make it PCA-like while taking advantage of non-linearity from autoencoders. A probabilistic interpretation with variational AE (VAE) is then provided and we integrated the new implementation with two existing methods on a new synthetic dataset 'Rectangles'. We also propose a classification downstream task as a quantitative metric alongside the standard analysis of covariance and interpolations.

1 Introduction

In modern ML practices, a high number of dimensions leads to a variety of issues: higher chance for overfitting due to the stronger assumptions, higher model complexity, stronger noise, and more computational power. Dimensionality reduction algorithms not only provides a pathway to removing the redundant features and improving the overall performance, but it also helps with data visualization for more intuitive explanations [1]. Two main categories stem from the large family of approaches - feature selection and feature engineering. While the former identifies and selects the relevant features for the dataset, the latter creates "new features" based on the existing ones manually.

Principal Components Analysis(PCA)[2], an imperative method in feature engineering, is used to drop the least important variables while keeping the more informative ones. For data points scattered in an N-D space, some axes(directions) can be identified with larger amount of variance that are orthogonal to each other. These axes(principal components) are found by a sequence of standardization, covariance calculation, and eigenvalue-eigenvector calculation steps. Given an arbitrarily selected number of principal components M, the algorithm can output the new set of mutually orthogonal features in space $\mathbb{R}^M$ for each data point with ordered importance. One main disadvantage to this method is the reduced interpretability due to the new feature space linearly transformed from the original space.

Similar to PCA, autoencoders can also be used to encode the input features into a more compact representation in the latent space. However, autoencoders are not restricted to linear transformations in their encoder/decoder networks, which implies an enhanced capability in capturing important non-linear features often found in more complex datasets (such as most vision and language datasets). For instance, with an MNIST handwritten digit database [3], we could expect a principal axis along the stroke direction and another along the 'thickness' dimension. The stroke shape axis is most likely non-linear and therefore it's not possible to be captured by a linear method such as PCA.

Preprint. Under review.

However, the naive autoencoder solution leaves two important characteristics behind: (1) ordering of features by importance and (2) independence between each dimension. In order to both leverage the structural complexity of the autoencoders and avoid the listed issues, multiple PCA-like autoencoding mechanisms have been developed [4, 5] (See section 2). Instead of producing the latent space representations simultaneously, both methods adopted a sequential trick to emphasize the importance of some units to achieve an equivalent ordering and independence.

Since almost all existing methods employ simple autoencoders as the base model, we then propose to explore variational autoencoders(VAE) [6] from a probabilistic perspective for learning a well-behaved latent representation. Intuitively, we can think of VAE as a regularized autoencoder to reduce overfitting and therefore to ensure the smoothness and continuity in latent space for generation and other downstream tasks. More importantly, We also provide a probabilistic interpretation (from which the intuition above can be stemmed) and integrate it with the existing methods.

Hence, the main contributions of this paper are:

1. Creating a synthetic dataset for better understanding of the properties of latent representations (Section 3)
2. Demonstrating the difference between PCA-encodings and autoencoder-encodings in the latent space (Section 4)
3. Reproducing the two PCA-like autoencoders implementations and evaluate on our own dataset for qualitative and quantitative comparisons (Section 5)
4. Integrating VAE into the PCA-like autoencoding procedures and illustrate its significance (Section 6)

Furthermore, in addition to visual illustrations and losses for evaluation, we also come up with a metric-based on classification as a downstream task to showcase the effectiveness of the learned representations from a more quantitative perspective.

2 Related Work and Methods

PCA has been utilised for dimensionality reduction by creating new linear combinations of variables characterising the data since the 1960s [2], and this unsupervised algorithm can identify the underlying patterns and main axes in data such that the similarities and differences can be highlighted.

In the previous work [5], the ordering of importance was effectively achieved by training a sequence of autoencoders with increasing latent space size. Each step is trained with one extra dimension in latent space while keeping the previous units unchanged. The independence on latent encodings was achieved with an additional loss term under normalization conditions: $\mathcal{L}_{cov}(X) = \frac{1}{M}\sum_{i=1}^{k-1}\sum_{j=1}^{M} z_i^j z_k^j$, so that the covariance between each of the distinct components is minimized during training.

Another similar but more rigorously proven approach [4] uses dropout on units in the representation space with a prior distribution p_B to select the cutoff index in each iteration of training. Therefore, the nested dropout problem can be formulated as $(\hat{\theta}, \hat{\phi}) = argmin_{(\theta,\phi)}\mathcal{L}(\theta, \phi)$, where the objective function can be written as $\mathcal{L}(\theta, \phi) = \mathbb{E}_B[\mathcal{L}_{\downarrow b}(\theta, \phi)]$, with b drawn from the distribution $p_B(\cdot)$. The truncated objective $\mathcal{L}_{\downarrow b}(\theta, \phi)$ can be written as the mean of reconstruction cost for all samples with the last $K - b$ units removed, where K is the latent representation dimension:

$$\mathcal{L}_{\downarrow b}(\theta, \phi) = \frac{1}{N}\sum_{n=1}^{N} F(y_n, \hat{y}_{n\downarrow b})$$

.

The intuition behind this approach is that nesting can lead to an inherent ordering of importance of latent units since one particular unit always depends on those preceding it. The prior is chosen to be $p_B(b) = \rho^{b-1}(1-\rho)$, a geometric distribution with exponential decay. In the original work, it was shown that the non-linear autoencoder with nested dropout does not compromise the autoencoder optimal solution and that with some additional restraints on orthogonality, PCA solution can be exactly recovered.

The probabilistic framework of variational autoencoders (VAE) [6] provides a strong foundation for understanding the ad-hoc reconstruction losses as well as regulating the behaviour of the latent space units, by encoding the representations as probability distributions from some priors (normal distribution for simplicity). Extensions to VAE include β-VAE [7], Conditional VAE [8], Importance-weighted VAEs (IWAE) [9], etc., which essentially renders more flexibility in regularization.

3 Dataset

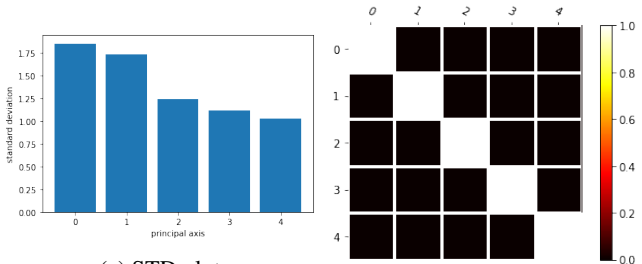
For different demonstration purposes, we proposed 2 synthetic datasets for the following experiments. The first one is a simple N-dim dataset `ClusteredPoints` generated from `sklearn.datasets.make_classification` API [10]. We specified 10000 samples and 3 classes with 5 informative and 5 redundant features out of 20 in total. This highly artificial and straightforward dataset is very convenient in identifying some key differences between different approaches, but we couldn't see how varying certain features can affect the generated results with clear visuals.

Figure 1: Examples of rectangles with different shape and size in `Rectangles` dataset

Therefore, the second dataset `Rectangles` amends this issue with a collection of rectangles with random sizes and shapes centred on the image. We generated 10000 images of shape 32×32 with a maximum greyscale of 255. A Gaussian noise is applied onto images (which is only visible on the edges) for data augmentation. Some examples can be seen in fig. 1.

4 Comparison of PCA and vanilla Autoencoder

The first experiments were conducted on the `ClusteredPoints` dataset to showcase the disparity of PCA results and vanilla autoencoders' latent encodings. We set the latent dimenstion as 5 for both algorithms, and the vanilla autoencoder is constructed symmetrically with only one linear layer on both decoder and encoder sides, trained with a simple cross-entropy reconstruction loss[11]. In PCA, the significance of a principal component axis is indicated by the variance, where we can see from fig. 2a that the importance for latent units is dropping consistently. We also display the covariance matrix (shown in fig. 2b) for all 5 encoded features, and all off-diagonal entries are close to 0, implying the orthogonality of the features.

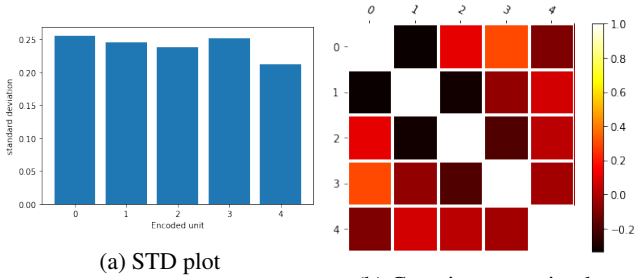


(a) STD plot

(b) Covariance matrix plot

Figure 2: Latent variable standard deviations and covariance matrix for PCA on `ClusteredPoints` dataset.

The same plots were shown for the linear autoencoder: (1) in fig. 3a, all 5 features show the same level of variance - that is, no importance priority is imposed on the units - and (2) in fig. 3b, most entries are non-zero, showing a strong correlation between features and hence, lack of independence. We further experimented with a few more intricate autoencoders (Multi-layer encoder-decoders, non-linear autoencoders) which led to the same conclusions (see fig. 13 in Appendix).



(a) STD plot

(b) Covariance matrix plot

Figure 3: Latent variable standard deviations and covariance matrix for vanilla autoencoder on `ClusteredPoints` dataset

5 Tailored Autoencoders

Having demonstrated the shortcomings of the autoencoders, we then turn to the PCA-like autoencoder implementation trained on the `Rectangles` dataset. The encoder is composed of 5 layers of convolutional layers with leaky ReLUs as the activation function for each, and the decoder is a symmetrical inverse of the encoder. Due to limited training resources, we randomly took a subset of the full training dataset of size 1000 for the subsequent trainings, and a batch size of 128, epoch number of 50, and learning rate at 5e-4 were set for all following experiments.

The ideal expectation was that through the independent representations, we can encode defining features of the rectangles in different dimensions. For instance, one unit may control the size and another the length-height ratio. We set the latent dimension to be 3 such that optimistically, 2 units can each capture a feature of interest, while the third unit is included for further observations. In order to visualise such individual effects, we only vary one unit in the range $[-1, 1]$ while keeping the rest unchanged at 0.

The baseline method shares the same encoder-decoder architecture but with a standard autoencoder training objective without any tricks. We show the interpolation results for each of the three units in fig. 4.

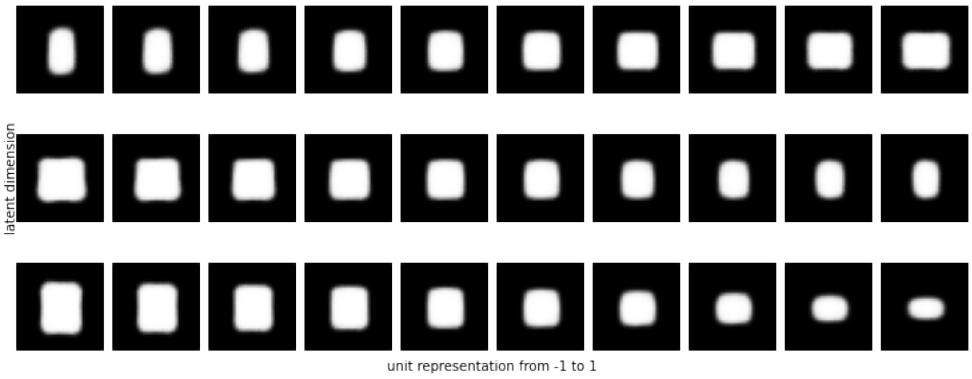


Figure 4: Interpolation of generated results by varying one dimension in latent space each time for vanilla autoencoder. No clear association of features can be seen.

We can see a clear transition of shape and/or size in all three units, but it is impossible to designate one feature to each, as they all show some similar changes. PCA-like autoencoder (`PCAAE`), in contrast, produced interpolation results in fig. 5. It is obvious that the dim-0 varies the size, dim-1 changes the width-height ratio, while dim-2 is redundant as no visibly significant changes occured along the interpolation plots. This straightforwardly solved the two issues of vanilla autoencoders: the importance of the first two dimensions evidently triumphed the last, resulting in an effective ordering, and each unit has an independent feature associated without interfering with others. We further demonstrate these points with covariance matrix and variance plots in fig. 6

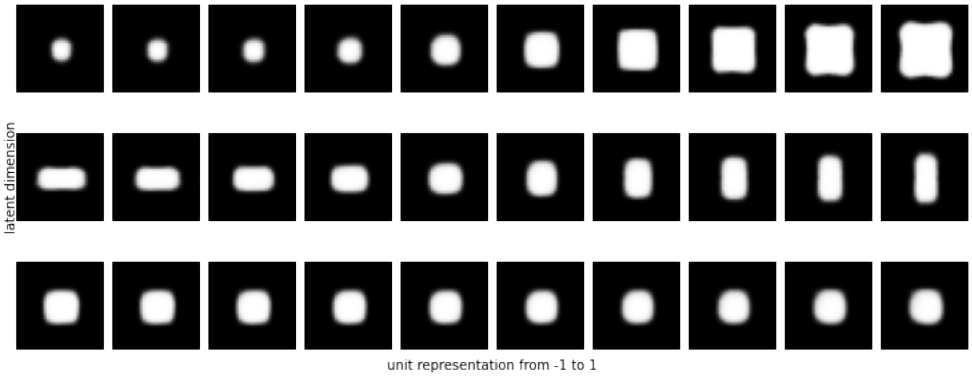
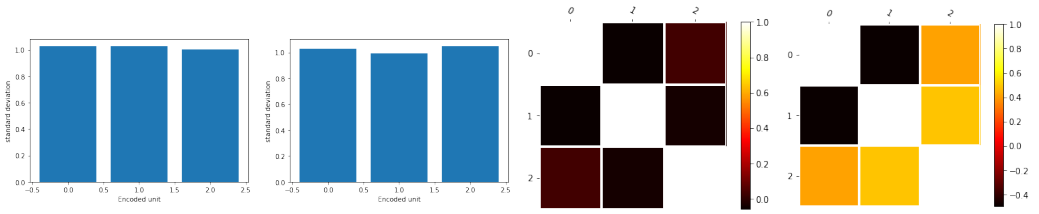


Figure 5: Interpolation of generated results by varying one dimension in latent space each time for PCAAE.



(a) PCAAE STD plot (b) AE STD plot (c) PCAAE covariance matrix plot (d) AE covariance matrix plot

Figure 6: Latent variable standard deviations and covariance matrix for vanilla autoencoder and PCAAE on `Rectangles` dataset

In fig. 6a and 6b, no visible ordering of variance can be observed, despite the clear differences shown from interpolations; however, fig. 6c showed near-zero covariances on off-diagonal entries, suggesting a strong assumption of orthogonality of latent features, as opposed to the baseline (fig. 6d).

We make the same plots for the nested dropout (`NDAE`) training procedure and compare with the previous two methods. In fig. 7, we see a very similar pattern in the three dimensions as the `PCAAE`, where the each of first two units are responsible for shape and size changes, while the last has minimal impact. This is further corroborated in fig. 8a, where the third latent unit saw a dip in variance. However, the off-diagonal covariance terms are non-zero as we did not impose regularization on latent space independence (fig. 8b).

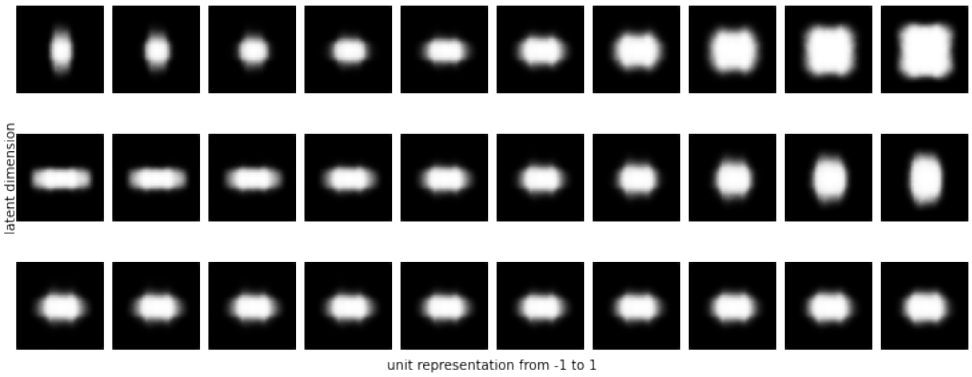
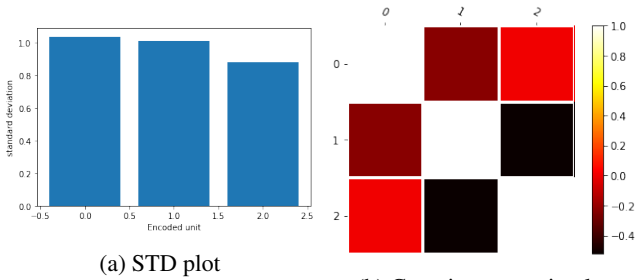


Figure 7: Interpolation of generated results by varying one dimension in latent space each time for NDAE.



(a) STD plot

(b) Covariance matrix plot

Figure 8: Latent variable standard deviations and covariance matrix for NDAE on `Rectangles` dataset

6 PCA-like VAEs

Having shown the properties of the PCA-like autoencoders, we then replace the autoencoder with VAEs, such that the new training objective is minimizing the log-likelihood of the generated data. Mathematically, we want to find θ that minimizes $\mathcal{L}(\theta) = \sum_i \log \mathbb{E}_{z\sim Z} Pr_X(x_i|Z = z;\theta)$ [12], where Z is the random variable of the latent space from which z is drawn. One trick for VAE is using importance sampling to estimate the expectation value, such that

$$\mathcal{L}(\theta) \approx \sum_i \log \left[\frac{1}{m} \sum_{j=1}^{m} \left\{ Pr_X(x_i|Z = z_j;\theta) \frac{Pr_Z(z_j)}{Pr_{\tilde{Z}}(z_j)} \right\} \right]$$

where z_j is sampled from $\tilde{Z}$, a prior distribution of choice.

With the encoder network ϕ which generates the sampling distribution $\tilde{Z}$ and Jenson's inequality, we can find a lower bound for the log-likelihood as $\mathcal{L}_{lb}(\theta,\phi) = \sum_i \mathbb{E}_F \log \left\{ Pr_X(x_i|Z = z_j;\theta) \frac{Pr_Z(\tilde{Z})}{Pr_{\tilde{Z}}(\tilde{Z})} \right\}$, where we can rewrite the effective loss as

$$\mathcal{L}_{lb}(\theta,\phi) = \sum_i \left\{ \left[\mathbb{E}_F \log Pr_X(x_i|Z = z_j;\theta) \right] - KL(Pr_{\tilde{Z}} \| Pr_Z) \right\}$$

We brought in F, a latent random variable as part of the reparameterization trick in the encoder [12], and the KL-divergence comes from its definition.

The first term can be interpreted as the effective "reconstruction loss", such that the first term decreases when data point x_i has a higher likelihood. The second term can be understood as the regularization to make $\tilde{Z}$ well-behaved, such that, for example, close points in the latent space also correspond to data of proximity in the original space.

In our implementation, we choose a multivariate Gaussian prior for the latent random variable for Z for convenience of KL divergence calculation, and we also assume that the generated random variable from the decoder also follows a MVN distribution. Hence, we subsequently replace the previous training objectives with the new negative-loglikelihood minimization shown above. We first show the interpolation results for each of the two methods(fig. 9, 10):

One apparent distinction from the previous interpolation plots was that VAE-generated rectangles tend to be more 'fuzzy' on the edges. This was expected as the Gaussian prior imposed on the generated space could lead to the blurring away from the centre. In addition, despite the smoother transition of generated images due to the KL divergence term, the orthogonality of the three latent dimensions is less evident and no importance ordering is visible as we slowly vary z in one dimension. The lack of training samples and epochs can be a major problem for this probabilistic approach. Nevertheless, it can still be observed from fig. 11b that the latent feature independence was preserved for PCA-VAE.

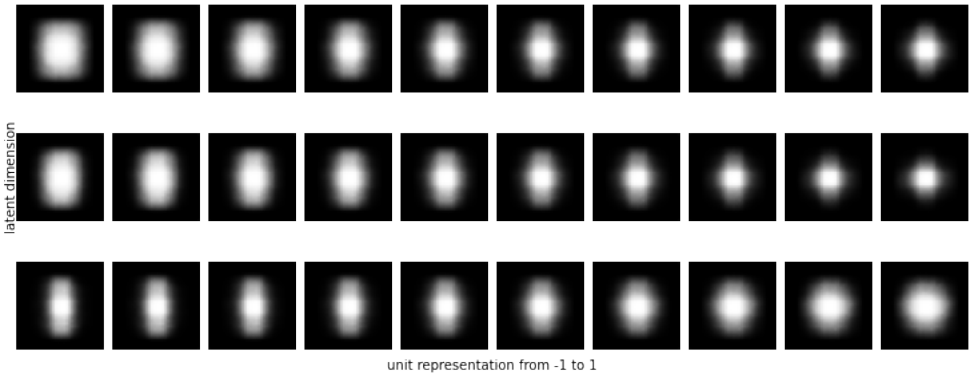


Figure 9: Interpolation of generated results by varying one dimension in latent space each time for PCA-VAE.

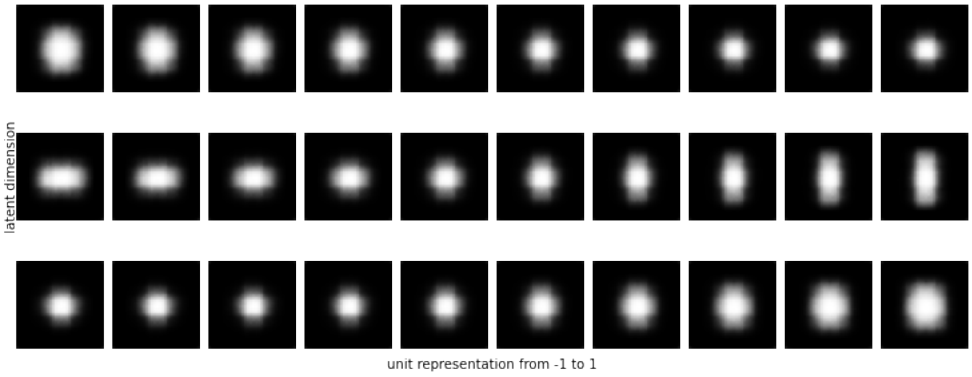
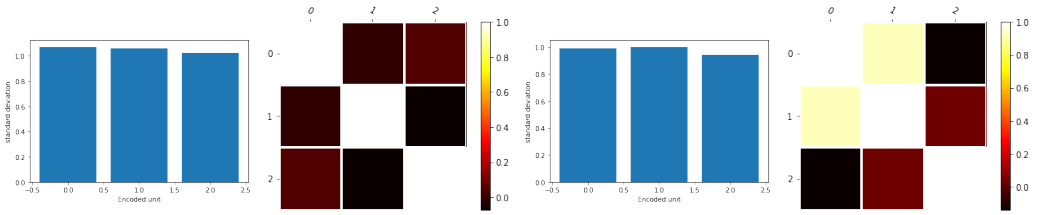


Figure 10: Interpolation of generated results by varying one dimension in latent space each time for ND-VAE.



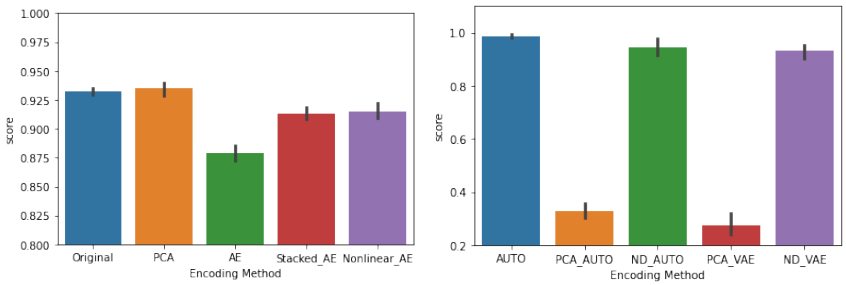
(a) PCA-VAE STD plot (b) PCA-VAE covariance matrix plot (c) ND-VAE STD plot (d) ND-VAE covariance matrix plot

Figure 11: Latent variable standard deviations and covariance matrix for NDAE on `Rectangles` dataset

7 Evaluation on classification downstream task

So far, we have been comparing the performances of different autoencoding approaches for PCA primarily based on interpolation plots as qualitative observations from which we drew valuable conclusions. However, since our autoencoder-based and VAE-based methods employed different training objectives - namely, minimizing reconstruction loss and negative log likelihood - it is challenging to directly compare the effectiveness quantitatively. We therefore propose a metric for easier and unified quantitative evaluation: 4-class-svm cross-validation score. As one of the many objectives of PCA is to use the representations of reduced complexity for further tasks, we, thus, divide the test `Rectangles` images into 4 groups based on size and shape and train a classifier on them with the latent features as inputs and the class division as the label. The correctness of this

approach can be verified from fig. 12a, where we benchmarked the performance of the representations obtained from different methods on a separate classification task (where the classes are pre-defined in the randomly generated synthetic data). We can find that PCA slightly outperforms the classification based on original data, while the autoencoders slightly worse. It was also expected that the stacked and non-linear autoencoders have slightly higher cross-validation scores than the shallow vanilla AE. We then apply the same evaluation to the rectangle dataset and the approaches demonstrated above. Although both PCAAE and NDAE showed promising results in the generative task with improved importance ordering and independence, neither surpassed the accuracy of vanilla autoencoder, and most notably, PCAAE saw a close-to-random-guess performance (see fig. 12b).



(a) `ClusteredPoints` evaluation

(b) `Rectangles` evaluation

Figure 12: Quantitative Evaluation of PCA and PCA-like methods on 4-way classification task. The score displayed is the average of 5 cross-validation accuracies.

One plausible explanation is that the holistic character of the latent features is lost when encoding the variables one at a time with encoder output of size one in PCAAE. Such individual representations, though pertinent to generative tasks, may lose the features needed for classification tasks based on the latent representation as a whole.

8 Future Work

In our experiments, we trained our adapted models on a smaller subset (1000 samples) of the full `Rectangles` dataset(10000 samples) due to limited training resources. A full training with all samples could further improve the current results. In addition, we arbitrarily set the number of latent dimensions to be three, and therefore it is expected to bring more insights by varying the number of latent variables and observe the impact on different metrics. Furthermore, our PCA-VAE method did not perform well on the classification downstream task, so that an improved method of reducing the 'discreteness' among latent variables can be designed. Lastly, the quantitative evaluation task specific to the `Rectangles` dataset can be a more fine-grained classification or even a regression task for further explorations.

9 Conclusion

Using the synthetic dataset of rectangles of different shapes and sizes, we first demonstrated that the modified autoencoders based on variable-by-variable encoding and nested dropout on latent space can both produce PCA-like latent variables with one or both of the key features - importance ordering and orthogonality. In both cases, the first two variables encode the features with the third one redundant. We then integrated VAE with the two approaches and presented a probabilistic interpretation of PCA-like autoencoding, along with the qualitative results, which appeared to be weaker than those before the integration largely due to lack of training. Finally, an original evaluation metric based on 4-class classification was proposed and measured on all experiments, showing that the original vanilla autoencoder and Nested Dropout-enriched methods are more competent in capturing distinctive features for classification tasks.

References

[1] Laurens van der Maaten, Eric Postma, and H. Herik. Dimensionality reduction: A comparative review. *Journal of Machine Learning Research - JMLR*, 10, 01 2007.

[2] Andrzej Maćkiewicz and Waldemar Ratajczak. Principal components analysis (pca). *Computers & Geosciences*, 19(3):303–342, 1993.

[3] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition, 1998.

[4] Oren Rippel, Michael A. Gelbart, and Ryan P. Adams. Learning ordered representations with nested dropout, 2014.

[5] Saïd Ladjal, Alasdair Newson, and Chi-Hieu Pham. A pca-like autoencoder, 2019.

[6] Diederik P Kingma and Max Welling. Auto-encoding variational bayes, 2014.

[7] Irina Higgins, Loïc Matthey, Arka Pal, Christopher P. Burgess, Xavier Glorot, Matthew M. Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

[8] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.

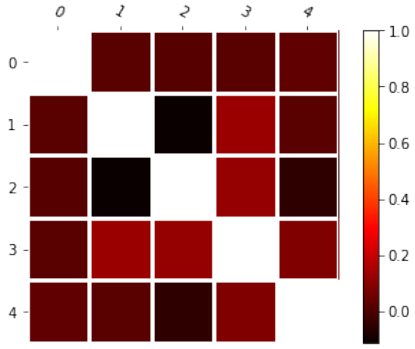
[9] Yuri Burda, Roger Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders, 2016.

[10] Isabelle Guyon. Design of experiments for the nips 2003 variable selection benchmark. 2003.

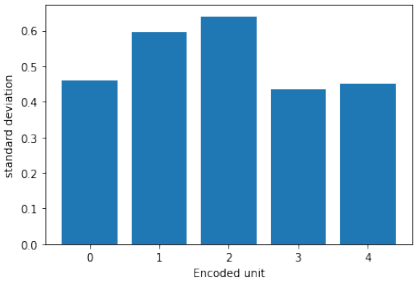
[11] Andrea Castiglioni. Dimensionality reduction with autoencoders versus pca, Apr 2021.

[12] Damon Wischik. Probabilistic machine learning notes, 2021.

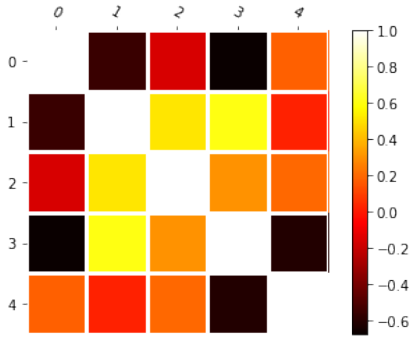
A Appendix



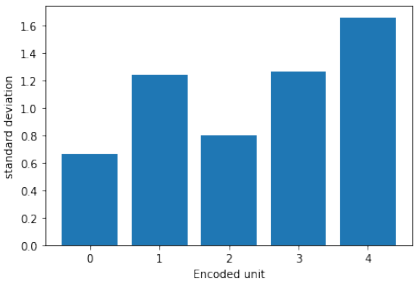
(a) stacked-autoencoder covariance matrix



(b) stacked-autoencoder STD plot

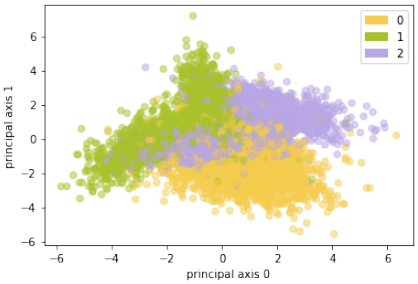


(c) nonlinear-autoencoder covariance matrix

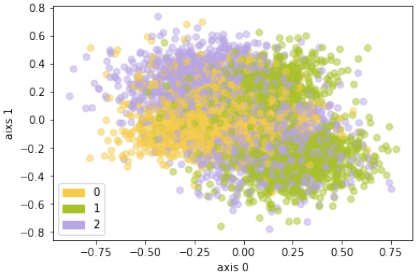


(d) nonlinear-autoencoder STD plot

Figure 13: Latent variable standard deviations and covariance matrix for stacked and non-linear autoencoders on `ClusteredPoints` dataset



(a) PCA clusters



(b) Autoencoder clusters

Figure 14: Illustration of datapoints of different classes from `ClusteredPoints` projected onto the subspace defined by the first two axes (out of 5)