UNIVERSIDADE FEDERAL DE PERNAMBUCO

Anthonny Dayvson Lino Paz

Exploring the Latent Space: Compact Representations with Autoencoders

Recife

2025

Anthonny Dayvson Lino Paz

Exploring the Latent Space: Compact Representations with Autoencoders

Relatório final, apresentado a Universidade Federal de Pernambuco, como parte das exigências para a obtenção do título de bacharel em Ciência da Computação.

Recife: 14 de agosto de 2025

Orientador (a): Tsang Ing Ren

Recife

2025

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Paz, Anthonny Dayvson Lino.
Exploring the latent space: compact representations with autoencoders / Anthonny Dayvson Lino Paz. - Recife, 2025.
17 p : il., tab.

Orientador(a): Tsang Ing Ren
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Ciências da Computação - Bacharelado, 2025.
8.
Inclui referências.

1. Autoencoders. 2. Classification. 3. Data Reconstruction. 4. Dimensionality Reduction. 5. Latent Space. I. Ren, Tsang Ing. (Orientação). II. Título.

000 CDD (22.ed.)

Anthonny Dayvson Lino Paz

Exploring the Latent Space: Compact Representations with Autoencoders

Relatório final, apresentado a Universidade Federal de Pernambuco, como parte das exigências para a obtenção do título de bacharel em Ciência da Computação.

Recife: 14 de agosto de 2025

BANCA EXAMINADORA

_____

Prof. Tsang Ing Ren (Orientador)

Universidade Federal de Pernambuco(UFPE), Centro de Informática (CIn)

_____

Prof. George Darmiton (Examinador Interno)

Universidade Federal de Pernambuco(UFPE), Centro de Informática (CIn)

Exploring the Latent Space: Compact Representations with Autoencoders

Anthonny D. Lino Paz, Tsang Ing Ren
Centro de Informática (CIn) – Universidade Federal de Pernambuco (UFPE)
Recife – PE – Brasil
adlp@cin.ufpe.br

Abstract

This work analyzes how latent-space size and bottleneck structure affect reconstruction quality and downstream utility in convolutional autoencoders. We implement and evaluate four variants—standard (`conv_ae`), sparse (`conv_sparse`), denoising (`conv_denoising`) and variational (`conv_vae`)—on CIFAR-10 across five latent dimensions (16, 32, 64, 128, 256). Reconstructions are assessed with MSE, SSIM, PSNR, ERGAS and UQI; latent embeddings are evaluated with supervised classifiers (Logistic Regression, MLP, Random Forest, KNN) and unsupervised clustering (KMeans, GMM, HDBSCAN) using ARI and NMI. Results indicate that `conv_sparse` attains the best perceptual reconstruction scores (e.g., MSE ≈ 0.0021, SSIM ≈ 0.89 at $d = 256$), `conv_denoising` yields the most discriminative embeddings for classification (best MLP accuracy ≈ 0.5471 at $d = 256$), and `conv_vae` underperforms at small d. Unsupervised clustering recovery is weak (ARI/NMI typically < 0.2), motivating future work on contrastive and clustering-aware objectives and modified VAE losses.

Keywords: Autoencoders; Classification; Data Reconstruction; Dimensionality Reduction; Latent Space

1 Introduction

The success of deep learning in computer vision owes much to the ability of neural networks to learn compact, semantically meaningful representations of data. Among the most versatile tools in this regard are **Autoencoders**—neural architectures designed to reconstruct inputs through a compressed latent representation. Their use spans tasks from dimensionality reduction (HINTON; SALAKHUTDINOV, 2006) and anomaly detection (AN; CHO, 2015) to generative modeling (KINGMA; WELLING, 2013) and pretraining (ERHAN et al., 2010).

The **latent space** is the core of an autoencoder. It encodes the high-dimensional input into a lower dimensional manifold, ideally retaining only essential semantic information. Yet, despite the centrality of the latent space, the impact of its size and structural regularization remains underexplored in systematic, comparative studies—especially in convolutional contexts. While some works address information bottlenecks (TISHBY; ZASLAVSKY, 2015) or visual disentanglement (HIGGINS et al., 2017), few quantify the trade-offs imposed by different latent dimensionalities and constraints under controlled experimental conditions.

This paper addresses this gap found in the literature. We conduct a thorough empirical investigation of the effects of latent space dimensionality and structure on reconstruction quality in convolutional autoencoders. Our methodology includes training four CAE (Convolutional Autoencoders) variants—standard, sparse, denoising, and variational—on the CIFAR-10 dataset across varying latent dimensions. We evaluate performance using both traditional and perceptually aligned metrics.

1.1 Contributions

- We implement and compare four convolutional autoencoder variants under consistent architectural and training regimes.
- We evaluate the effect of latent dimensionality using five latent space sizes (16, 32, 64, 128, 256, 512, 1024, 2048).
- We assess reconstruction quality with MSE, SSIM, PSNR and other metrics, ensuring both quantitative and perceptual evaluation.
- We provide a reproducible experimental framework and publicly available code repository.
- We discuss the implications for representation learning, model design, and interpretability in downstream tasks.

1.2 Objectives

The main goal of this work is to investigate how the dimensionality and structure of the latent space affect the reconstruction quality and semantic organization in convolutional autoencoders.

1.2.1 Specific objectives include:

- Implement and compare standard, sparse, denoising, and variational autoencoders under a consistent architecture.
- Evaluate reconstruction quality across different latent dimensions using perceptual and pixel-wise metrics.
- Analyze the discriminative capacity of latent embeddings using supervised classification.
- Assess latent space organization through visualization and clustering techniques.

2 Related Work

Autoencoders, introduced by Rumelhart et al. (RUMELHART; HINTON; WILLIAMS, 1986) and popularized in deep learning by Hinton & Salakhutdinov (HINTON; SALAKHUTDINOV, 2006), are widely used for unsupervised learning. Convolutional variants (CAEs) exploit the spatial locality of image data (MASCI et al., 2011), making them suitable for vision tasks.

Sparse Autoencoders (SAEs) encourage minimal activation in hidden units, promoting compact and disentangled representations (NG, 2011). Denoising Autoencoders (DAEs) (VINCENT et al., 2010) train the network to recover clean inputs from corrupted versions, improving robustness and generalization. Variational Autoencoders (VAEs) (KINGMA; WELLING, 2013) combine reconstruction with probabilistic latent modeling, making the latent space smooth and generative.

The structure of the latent space affects not only reconstructions but also interpretability (HIGGINS et al., 2017), robustness (ZHANG et al., 2018), and downstream task performance (BENGIO; COURVILLE; VINCENT, 2013). Studies like (BENGIO; COURVILLE; VINCENT, 2013) suggest that low-dimensional spaces often force generalization, but too aggressive compression can eliminate semantic content. Others have examined topology and continuity in the latent manifold (LOCATELLO et al., 2019; BALLE; LAPARRA; SIMONCELLI, 2017).

However, most prior works either focus on a single architecture or analyze performance qualitatively. Our study differs in scope and rigor by combining multiple model types, dimensional scales, and evaluation metrics under consistent conditions.

3 Methodology

3.1 Dataset

We use CIFAR-10 (KRIZHEVSKY, 2009), a well-established benchmark in image classification and reconstruction. It contains 60,000 color images of size 32x32 across 10 object categories, with 50,000 used for training and 10,000 for testing. Images are normalized to [0, 1] and optionally augmented for robustness analysis.

3.2 Model Architectures

All models share the same convolutional backbone (`ConvEncoder` + bottleneck linear layer + `ConvDecoder`); they differ only in their bottleneck behavior and loss regularization. The encoder and decoder components are implemented as the `ConvEncoder` and `ConvDecoder` modules, respectively.

- **Backbone:** In deep learning, the term *backbone* refers to the core feature extractor shared across different models. In our case, it comprises the encoder architecture followed by a linear projection layer (latent bottleneck) and the decoder.
- **Encoder:** The encoder performs hierarchical feature extraction via stacked convolutional blocks that progressively increase the receptive field and spatial abstraction. Each block applies convolution, normalization and a nonlinearity, with strided downsampling used to condense information into lower-resolution feature maps; the final feature map is flattened and projected by a linear layer to produce the latent vector z. This design captures local and compositional image patterns across scales, yielding compact, semantically meaningful embeddings for downstream analysis.
- **Latent Bottleneck:** This is the compressed representation of the input data in a low-dimensional latent space $\mathbb{R}^d$. It serves as the information bottleneck where the encoder output is projected before being passed to the decoder. The structure and dimensionality of this bottleneck directly influence the recosntruction performance and representation quality.
- **Decoder:** The decoder maps the latent representation back into the original input space. It is implemented via a stack of transposed convolutional layers that upsample the feature maps, recosntructing the spatial dimensions of the original image.

The following model variants are explored (code available at GitHub).

a. **Standard Convolutional Autoencoder (ConvAE)**

- **Encoder:** The `ConvEncoder` consists of six convolutional blocks, each implemented as `Conv2d + BatchNorm2d + ReLU`. Downsampling is achieved via stride-2 convolutions in the 2nd, 4th, and 6th layers.
- **Latent bottleneck:** The final feature map of size $256 \times 4 \times 4$ is flattened and passed through a linear projection layer (`fc_enc`) to obtain a latent vector $z \in \mathbb{R}^d$.
- **Decoder:** The `ConvDecoder` projects z back to the original input size. It includes a linear layer that reshapes the latent code to $256 \times 4 \times 4$, followed by three transposed convolutional blocks and a final `Conv2d`(3,3,1,1) with a Sigmoid activation.
- **Forward pass:** The standard forward pass involves encoding an input x to obtain z, and decoding z to reconstruct x.

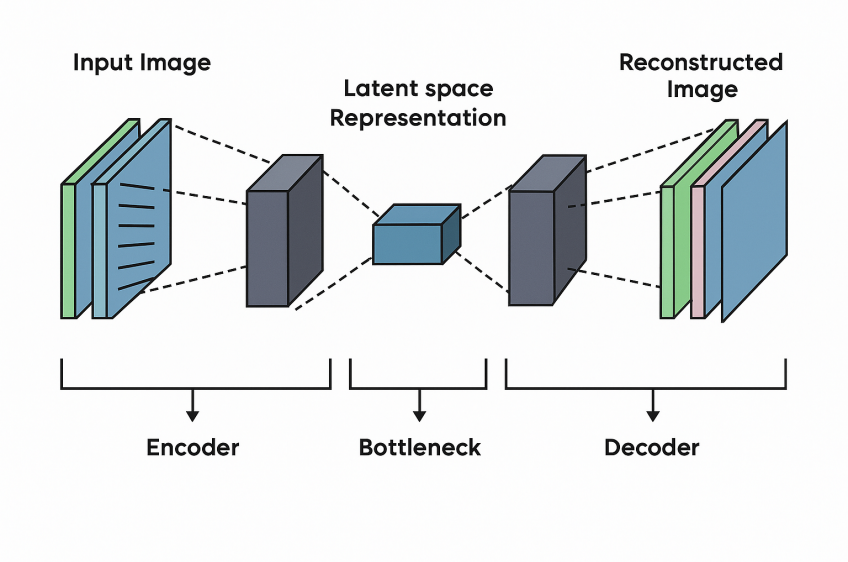


Figure 1: Standard Convolutional Autoencoder Architecture

b. **Sparse Convolutional Autoencoder (Sparse ConvAE)**

- Inherits the same backbone as ConvAE.
- **Latent bottleneck:** Same as above; however, an L_1 penalty is applied to the latent vector z during training to promote sparsity in the learned representation.
- **Motivation:** Sparse representations are encouraged to improve interpretability and reduce redundancy in the latent space.

c. **Denoising Convolutional Autoencoder (Denoising ConvAE)**

- Architecturally identical to ConvAE.
- **Forward pass:** During training, the input image x is corrupted with additive Gaussian noise (`noise_factor` $= 0.3$), and the model is trained to reconstruct the original clean image. This encourages the model to learn robust features that generalize better to noisy data.

d. **Variational Convolutional Autoencoder (ConvVAE)**

- Shares the same encoder and decoder modules, but replaces the deterministic latent bottleneck with a probabilistic one.
- **Latent bottleneck:** Two separate linear layers map the flattened encoder output to vectors μ and $\log \sigma^2$ in $\mathbb{R}^d$, representing the mean and log-variance of a learned Gaussian distribution.
- **Reparameterization:** To allow backpropagation through sampling, we compute $z = \mu + \exp(0.5 \log \sigma^2) \odot \epsilon$, where $\epsilon \sim \mathcal{N}(0, I)$. This transformation enables end-to-end optimization despite the stochastic nature of the sampling.
- **Loss function:** The total loss is composed of the reconstruction loss (e.g., MSE) and the Kullback–Leibler divergence between the approximate posterior $\mathcal{N}(\mu, \sigma^2)$ and the unit Gaussian prior $\mathcal{N}(0, I)$.

3.3 Latent Space Dimensions

We vary the size of the latent vector in {16, 32, 64, 128, 256, 512, 1024, 2048}, simulating undercomplete and overcomplete representations. The impact of latent space scale is analyzed per architecture and across metrics.

3.4 Training Setup

All models were implemented in Python using the PyTorch framework. Training was conducted on a machine equipped with an NVIDIA RTX 3050 GPU (12GB VRAM), Intel Core i5, and 8GB of RAM, running Windows 11. Each model was trained from scratch using the Adam optimizer with default parameters ($\beta_1 = 0.9$, $\beta_2 = 0.999$). The learning rate was set to 1×10^{-3} and batch size to 128. Training was performed for 50 epochs with early stopping based on training loss monitored over a patience of 5 epochs. The loss function used for standard, sparse, and denoising models was the Mean Squared Error (MSE) loss. For the sparse model, an additional L1 penalty was applied to the bottleneck activations with a regularization factor $\lambda = 1 \times 10^{-4}$. For the variational autoencoder (VAE), the total loss combined the reconstruction MSE with the Kullback-Leibler divergence (KL) between the approximate posterior and the standard normal prior. Gaussian noise with standard deviation $\sigma = 0.3$ was added to the input images in the denoising autoencoder during training. No data augmentation (e.g., flips or rotation) was used to isolate the effects of latent structure. Each experiment was repeated for eight different latent dimensions: 16, 32, 64, 128, 256, 512, 1024, 2048. All models were trained using a fixed random seed to ensure consistency across runs.

3.5 Evaluation Metrics

- **Mean Squared Error (MSE)**: Measures the average squared pixel-wise difference between the original and reconstructed image. Penalizes larger errors more strongly. Lower is better.
- **Root Mean Squared Error (RMSE)**: Square root of the MSE. Indicates the average magnitude of reconstruction errors in pixel intensity. Lower values denote better reconstruction quality.
- **Peak Signal-to-Noise Ratio (PSNR)**: Represents the ratio between the maximum possible pixel value and the power of reconstruction error. Higher values indicate better reconstruction fidelity and less noise.
- **Structural Similarity Index (SSIM)** (**WANG; BOVIK et al., 2004**): Evaluates perceptual similarity between original and reconstructed images based on luminance, contrast, and structural information. Values close to 1 indicate higher perceptual quality.
- **Universal Quality Index (UQI)** (**WANG; BOVIK, 2002**): Assesses structural fidelity by combining luminance, contrast, and correlation metrics into a single value.
- **Erreur Relative Globale Adimensionnelle de Synthèse (ERGAS)** (**WALD; RANCHIN; MANGOLINI, 2000**): Originally used in remote sensing, ERGAS estimates normalized global error between two images. Lower values imply higher reconstruction accuracy.
- **Adjusted Rand Index (ARI):** Measures the similarity between clustering results and ground truth labels, correcting for change. Values range from -1 to 1, where 1 indicates perfect clustering, 0 corresponds to random labeling, and negative values suggests anti-correlation.
- **Normalized Mutual Information (NMI):** Quantifies the mutual dependence between predicted clusters and true labels. It ranges from 0 (no mutual information) to 1 (perfect correlation).

4 Experiment and Results

This section presents a comprehensive evaluation of the proposed convolutional autoencoder architectures. We analyze quantitative reconstruction metrics, qualitative reconstructions examples, latent space visualizations, supervised classification results, and unsupervised clustering outcomes across different latent dimensions. The goal is to understand the interplay between latent space size and the performance and interpretability of each autoencoder variant.

4.1 Quantitative Evaluation

To evaluate the fidelity of image reconstructions, we use the metrics defined in Section 3.5. These were computed for each autoencoder variant—`conv_ae` (Standard Autoencoder), `conv_sparse` (Sparse Autoencoder), `conv_denoising` (Denoising Autoencoder), and `conv_vae` (Variational Autoencoder)—across eight latent dimensions: 16, 32, 64, 128, 256, 512, 1024, 2042.

Across all models, increasing the latent dimensionality improves reconstruction performance. The `conv_sparse` model achieved good results at $d = 256$, with MSE = 0.0021, ERGAS = 1.60, SSIM = 0.89, and PSNR = 27.29.

In contrast, `conv_vae` performed significantly worse at lower latent sizes. For instance, at $d = 16$, it yielded MSE = 0.0638, SSIM = 0.13, and PSNR = 12.59, indicating very blurry reconstructions. Its performance only approached the others at higher dimensional sizes.

These results confirm the critical roles of latent space capacity and architecture choice in determining reconstruction quality.

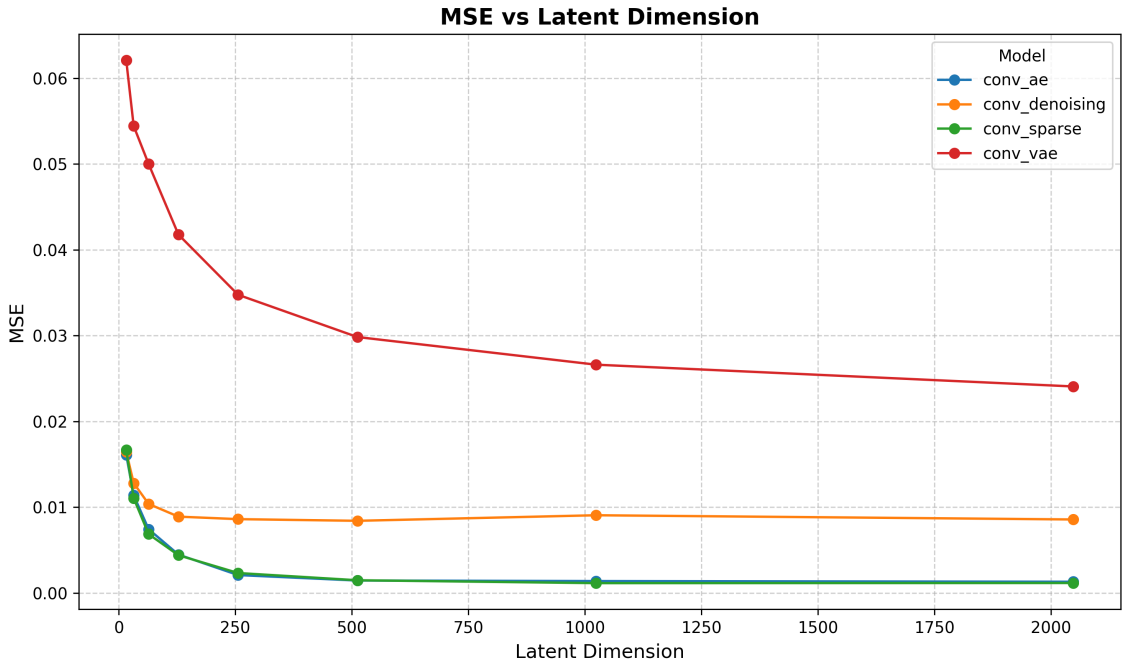


Figure 2: MSE for each model across latent dimensions.

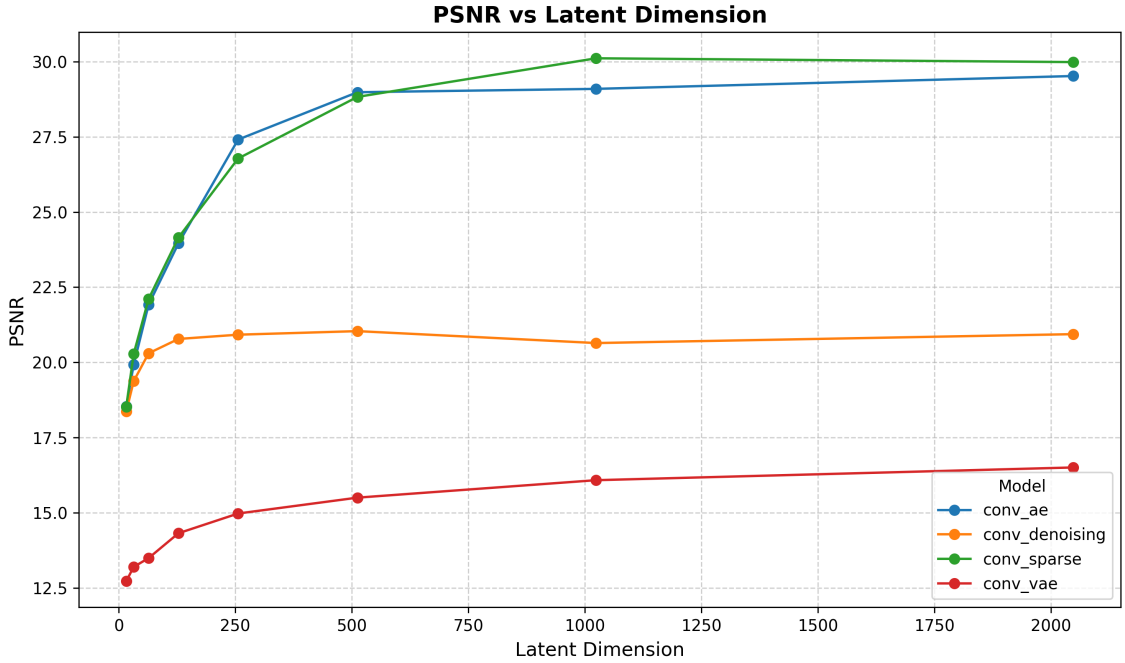


Figure 3: PSNR values indicating reconstruction quality across models.

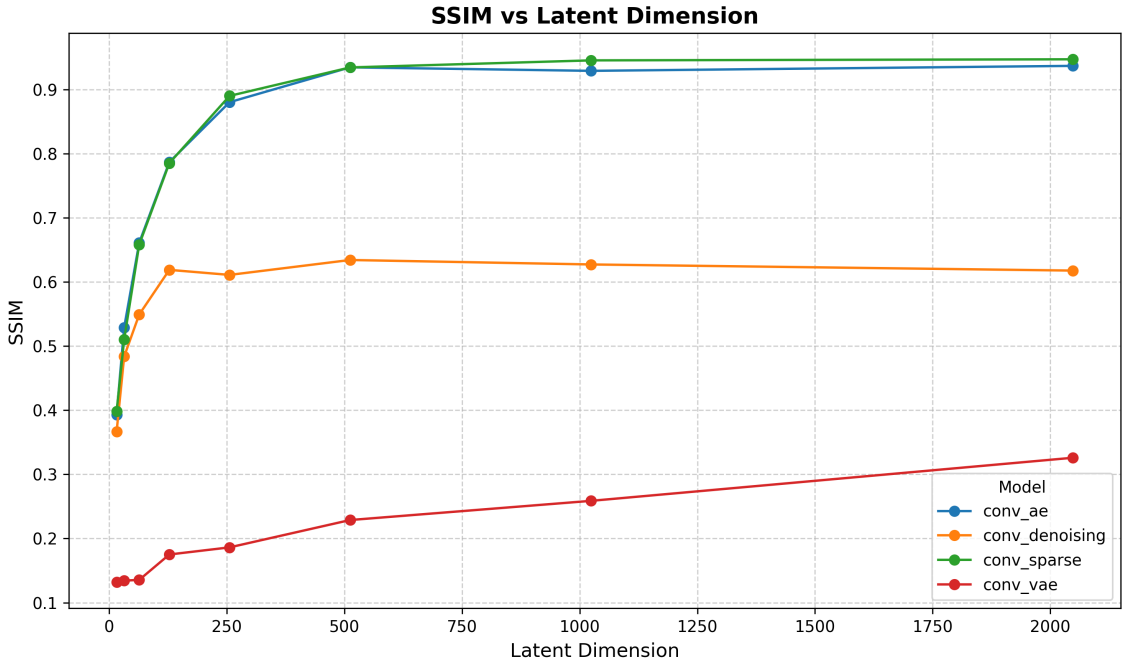


Figure 4: Perceptual similarity (SSIM) as a function of latent size.

Table 1: Quantitative reconstruction results (latent size = 256). Best values in bold.

Model	MSE	SSIM	PSNR	ERGAS	UQI
conv_ae	0.0025	0.88	26.74	3.44	0.98
conv_sparse	**0.0021**	**0.89**	**27.29**	**1.60**	**0.99**
conv_denoising	0.0089	0.62	20.74	4.00	0.98
conv_vae	0.0034	0.20	15.09	4.47	0.98

4.2 Qualitative Results

To complement the quantitative analysis, we show examples of reconstructed images for each model and latent dimension. These visualizations reveal perceptual differences not always captured by numerical metrics.

As expected, reconstructions improve visually as latent dimension increases. The conv_sparse and conv_ae models produce sharp and detailed images even at 64 dimensions, while the conv_vae model shows noticeable blurriness and loss of texture, particularly for small latent spaces.

The qualitative differences are aligned with SSIM and UQI scores, reinforcing their validity as perceptual quality indicators.

Figure 5: Original and reconstructed CIFAR-10 samples at conv_ae 64 dimensions.

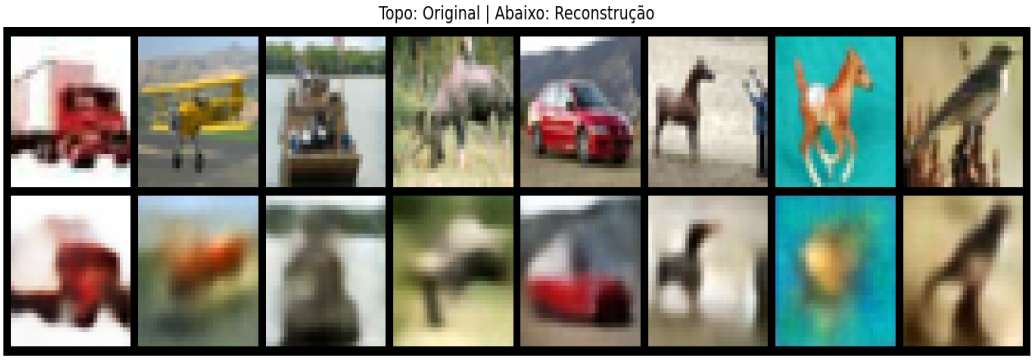


Figure 6: Original and reconstructed CIFAR-10 samples at conv_sparse 64 dimensions.

4.3 Latent Space Visualization

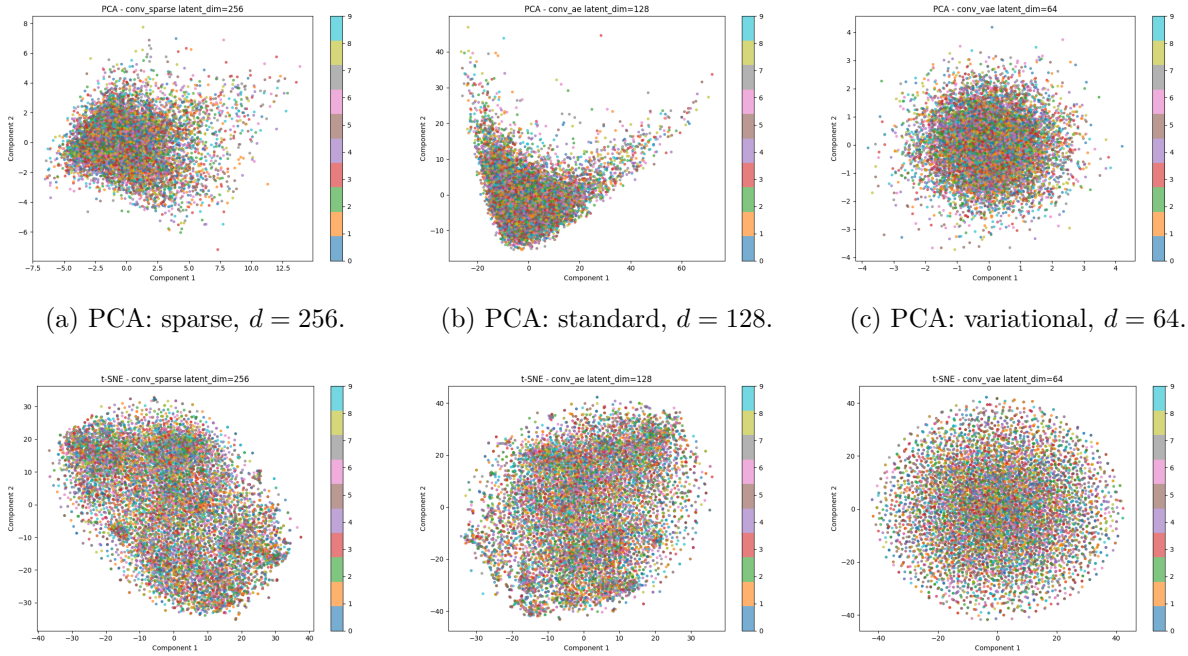
To assess how each autoencoder organizes semantic information, we project the learned embeddings into two dimensions using both Principal Component Analysis (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE). PCA captures global linear variance, while t-SNE emphasizes local neighborhood structure on the underlying manifold. By comparing both projections, we obtain a more complete picture of latent-space geometry.

PCA Projections. Figure 7 shows the first two principal components of the test-set embeddings for conv_ae, conv_sparse, and conv_vae at representative dimensions. All models exhibit a largely isotropic cloud in PCA space, with only slight elongations along the first component. This indicates that linear variance alone does not separate CIFAR-10 classes distinctly in the bottleneck.

t-SNE Projections. Figure 7 presents t-SNE embeddings for the same models and dimensions. Unlike PCA, t-SNE reveals small, ephemeral clusters in deterministic autoencoders. In particular:

- **conv_sparse** ($d = 256$)—t-SNE uncovers localized groupings of same-class points, suggesting that sparsity encourages tighter semantic neighborhoods.
- **conv_ae** ($d = 128$)—clusters are less compact but still discernible, reflecting the model's ability to encode class information without explicit regularization.
- **conv_vae** ($d = 64$)—embeddings remain diffusely mixed, consistent with the VAE's emphasis on distributional smoothness over class separability.

Interpretation. The PCA results confirm that global axes of maximum variance do not align with semantic class boundaries in any model. In contrast, t-SNE highlights that deterministic architectures (standard and sparse ConvAE) can learn locally coherent clusters, particularly at higher latent dimensions. The absence of distinct clusters in the VAE—even where t-SNE is most sensitive—underscores its trade-off between reconstruction fidelity and latent-space regularity.



(a) PCA: sparse, $d = 256$. (b) PCA: standard, $d = 128$. (c) PCA: variational, $d = 64$.

(d) t-SNE: sparse, $d = 256$. (e) t-SNE: standard, $d = 128$. (f) t-SNE: variational, $d = 64$.

Figure 7: Latent-space projections of test-set embeddings. Top row: PCA (global linear structure). Bottom row: t-SNE (local neighborhood structure). Color denotes true CIFAR-10 class.

4.4 Analysis and Discussion

This section synthesizes the findings from all metrics and evaluations to provide a holistic understanding of each model's performance and the role of latent dimension.

4.4.1 Reconstruction Analysis

All models benefit from increasing latent space size. The conv_sparse model consistently outperformed others in MSE, SSIM, PSNR and ERGAS, indicating high-quality and compact reconstructions. conv_denoising offered competitive results but slightly inferior to conv_sparse.

The conv_vae model underperformed in all metrics except when latent size was large (>= 256), where regularization becomes less restrictive. This confirms that VAEs suffer from poor reconstructions at low capability due to the KL divergence constraint.

No overfitting was observed in any model—the steady improvement in all metrics across dimensions suggests generalization is preserved.

4.4.2 Supervised Classification Analysis

To quantify the discriminative power of the latent embeddings, we train four simple classifiers—Logistic Regression (LR), a single-hidden-layer MLP, Random Forest (RF) and a K-Nearest Neighbors (KNN) classifier on the CIFAR-10 test embeddings. Table 2 summarizes accuracy for each model and latent dimension.

Table 2: Classification accuracy on CIFAR-10 test embeddings.

Model	Classifier	16	32	64	128	256
conv_ae	LR	0.3544	0.3816	0.3935	0.4075	0.4179
	MLP	0.4659	0.5119	0.5216	0.5204	0.5168
	KNN	0.4288	0.4657	0.4341	0.3884	0.3379
	RF	0.4617	0.4694	0.4579	0.4484	0.4454
conv_de	LR	0.3442	0.3750	0.3852	0.4394	0.5361
	MLP	0.4630	0.5069	0.5163	0.5332	0.5471
	KNN	0.4170	0.4358	0.4288	0.4106	0.4200
	RF	0.4520	0.4639	0.4644	0.4657	0.4777
conv_sp	LR	0.3560	0.3831	0.3927	0.4125	0.4183
	MLP	0.4716	0.5100	0.5155	0.5158	0.5029
	KNN	0.4328	0.4539	0.4211	0.3575	0.2733
	RF	0.4601	0.4700	0.4566	0.4518	0.4417
conv_vae	LR	0.2678	0.2973	0.3439	0.3832	0.4455
	MLP	0.3297	0.3526	0.3886	0.4160	0.4601
	KNN	0.2764	0.2894	0.3057	0.3310	0.3786
	RF	0.3301	0.3386	0.3601	0.3891	0.4304

The supervised classification results indicates that deterministic autoencoder variants (conv_ae, conv_sparse, and conv_denoising) generally produce more discriminative embeddings that the variational model, but the performance is far from perfect. Across classifiers and latent sizes, accuracies mostly fall in the 0.30-0.55 range. The best observed result in our experiments is approximately 0.55 (for the denoising model with MLP at d=256), while many configurations-especially those using small latent sizes-yield accuracies near 0.35-0.45. The conv_vae embeddings are the weakest overall at small dimensions and improve with larger latent size, but they still do not match the deterministic models in our setup. These outcomes suggest that deterministic reconstruction objectives produce embeddings with more class-discriminative information that the VAE's probabilistic bottleneck under the architectures and training regimen applied here; however, none of the evaluated models produce near-perfect linear separability on CIFAR-10 embeddings in this protocol.

4.5 Classification Pipeline

The classification stage was designed to systematically evaluate the discriminative capacity of the learned embeddings produced by each autoencoder variant. This pipeline comprises five main stages: extraction of embeddings, shape considerations, preprocessing, classifier instantiation, and final training/evaluation.

Data and Embeddings Extraction. For each model and latent dimension combination, the trained autoencoder is loaded and the `extract_embeddings` procedure is executed over both the training and test dataloaders. The model is set to evaluation mode and all computations are performed without gradient tracking. In deterministic autoencoders, the encoder outputs

$(h_{\text{conv}}, h_{\text{flat}})$, which are projected to the latent vector z through the fully connected encoder layer (`fc_enc`). In variational autoencoders (VAE), both the mean vector z_μ and the log-variance vector $z_{\log \sigma^2}$ are computed, and the latent representation is formed by their concatenation:

$$z = \text{concat}(z_\mu, z_{\log \sigma^2}, \text{dim} = 1).$$

Additionally, an intermediate pooled feature h_{pooled} is obtained via mean pooling over the convolutional feature maps. This pooled feature is concatenated with z to create the final *combined feature* vector. All embeddings and their corresponding labels are stored in serialized `.pt` files for subsequent processing.

Shape Considerations. In VAEs, the latent vector z has dimensionality equal to $2 \times$ latent_dim due to the concatenation of mean and log-variance. In deterministic models, z retains a dimensionality equal to latent_dim. This architectural difference results in different feature dimensionalities that must be handled consistently during preprocessing.

Preprocessing. Before classification, embeddings undergo normalization or transformation according to one of the following strategies:

- **Standard Scaling**: zero mean and unit variance.
- **Robust Scaling + PCA**: scaling robust to outliers, followed by Principal Component Analysis.
- **Min–Max Scaling + Feature Selection**: scaling to $[0, 1]$ interval, followed by `SelectKBest` with ANOVA F-test.

All preprocessing transformations are fit exclusively on the training set and applied to the test set to prevent data leakage. The best-performing preprocessing method for each classifier is determined based on the highest test accuracy.

Classifiers. The following classifiers are evaluated, with hyperparameters optimized via `GridSearchCV` (3-fold cross-validation):

- **Logistic Regression**: solver = `liblinear`, `max_iter` = 1000.
- **MLPClassifier**: hidden layers = (128, 64, 32), early stopping enabled.
- **Random Forest**: `n_estimators` = 200, `max_depth` = 20.
- **K–Nearest Neighbors**: $k = 5$, `weights` = `distance`.

Training and Evaluation. For each preprocessing strategy:

1. Perform hyperparameter tuning on the processed training set.
2. Compute mean and standard deviation of 5-fold cross-validation accuracy scores.
3. Retrain the classifier on the full training set using the selected hyperparameters.
4. Evaluate on the held-out test set.

For each model/latent size combination, the classification results are recorded as {*Test Accuracy*, *CV Mean*, *CV Std*} for every classifier. The highest test accuracy observed across all preprocessing–classifier pairs is reported as the final supervised classification score for that embedding configuration.

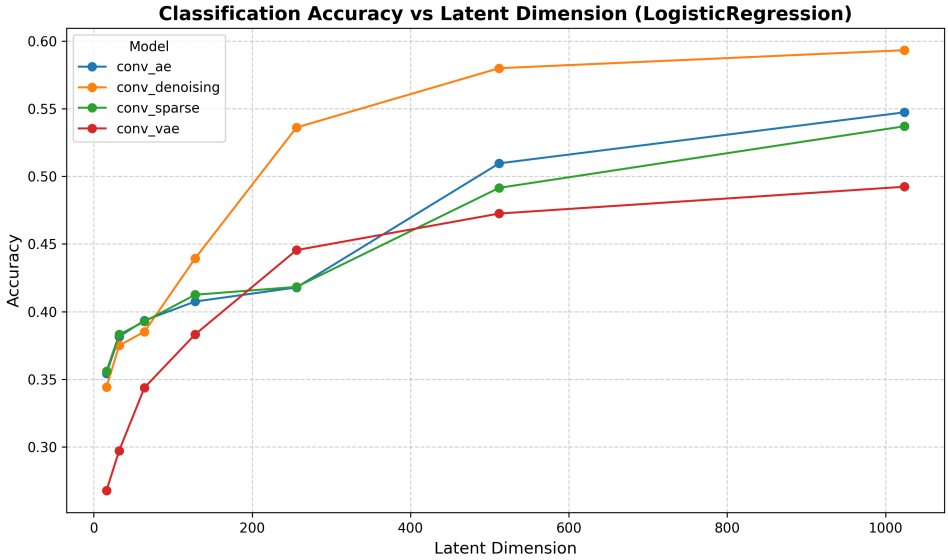


Figure 8: Logistic Regression accuracy across latent dimensions for each autoencoder variant.

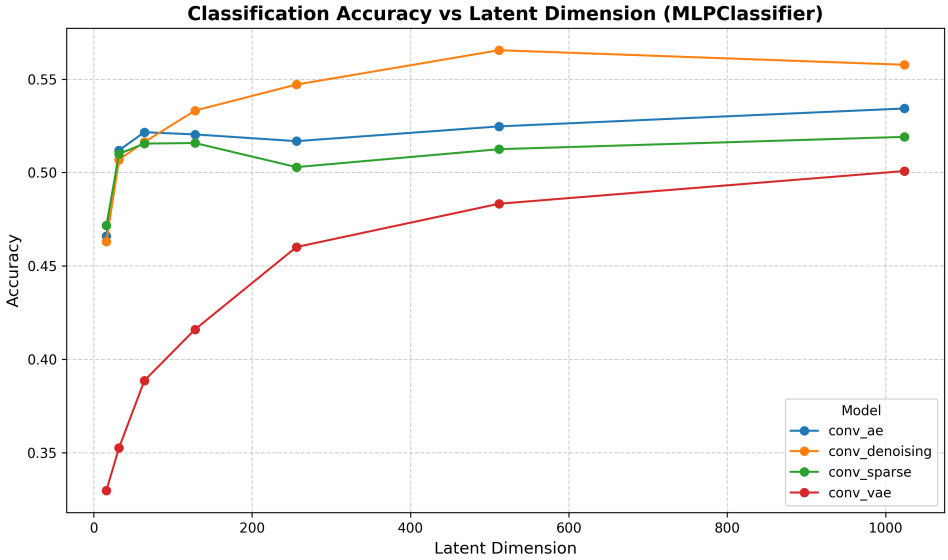


Figure 9: MLP Classifier accuracy across latent dimensions for each autoencoder variant.

4.5.1 Unsupervised Clustering Analysis

We evaluate whether embeddings naturally form clusters without label information. We apply K-Means, Gaussian Mixture Models (GMM), and HDBSCAN to each set of latent vectors and compute the Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI). Results are summarized in Table 3, and the K-Means and GMM metrics are visualized in Figure 10.

Table 3: Unsupervised clustering metrics (ARI, NMI) on latent embeddings (test split).

Model	**Method**	**Latent Dim.**				
		16	32	64	128	256
conv_ae	KMeans	(0.06, 0.11)	(0.06, 0.11)	(0.06, 0.11)	(0.05, 0.10)	(0.06, 0.11)
	GMM	(0.07, 0.13)	(0.08, 0.14)	(0.11, 0.17)	(0.09, 0.15)	(0.10, 0.16)
	HDBSCAN	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)
conv_sparse	KMeans	(0.06, 0.11)	(0.06, 0.11)	(0.06, 0.10)	(0.05, 0.09)	(0.03, 0.06)
	GMM	(0.07, 0.12)	(0.10, 0.16)	(0.08, 0.14)	(0.08, 0.14)	(0.08, 0.14)
	HDBSCAN	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)
conv_denoising	KMeans	(0.05, 0.09)	(0.04, 0.08)	(0.05, 0.09)	(0.04, 0.08)	(0.04, 0.07)
	GMM	(0.06, 0.11)	(0.06, 0.12)	(0.07, 0.13)	(0.06, 0.12)	(0.06, 0.12)
	HDBSCAN	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)	(0.00, 0.00)
conv_vae	KMeans	(0.03, 0.06)	(0.03, 0.07)	(0.03, 0.07)	(0.02, 0.06)	(0.04, 0.10)
	GMM	(0.04, 0.08)	(0.04, 0.07)	(0.05, 0.08)	(0.04, 0.08)	(0.05, 0.09)
	HDBSCAN	(0.00, 0.01)	(0.00, 0.03)	(0.00, 0.01)	(0.00, 0.00)	(0.00, 0.01)

Each entry is (ARI, NMI). Values are computed on the test split and rounded to two decimal places. Note: The modest GMM gains suggest that soft, Gaussian-like class structure exists in some

Table 3 reports ARI and NMI values that are low but non-zero across models and latent dimensions. KMeans achieves $ARI \approx 0.03$–0.06 and $NMI \approx 0.056$–0.114, while Gaussian Mixture Models (GMM) attain higher agreement (ARI up to ≈ 0.105; NMI up to ≈ 0.172, observed for `conv_ae` at $d = 64$). HDBSCAN yields virtually no meaningful clusters in most runs.

These numbers indicate that, although reconstruction-only training does not produce well-separated, high-quality clusters, there exists a weak, recoverable grouping signal (better captured by GMM than by KMeans). Thus, supervised classifiers can exploit class information that is present in the embedding geometry, even when that information is not organized as compact clusters readily recovered by off-the-shelf clustering methods.

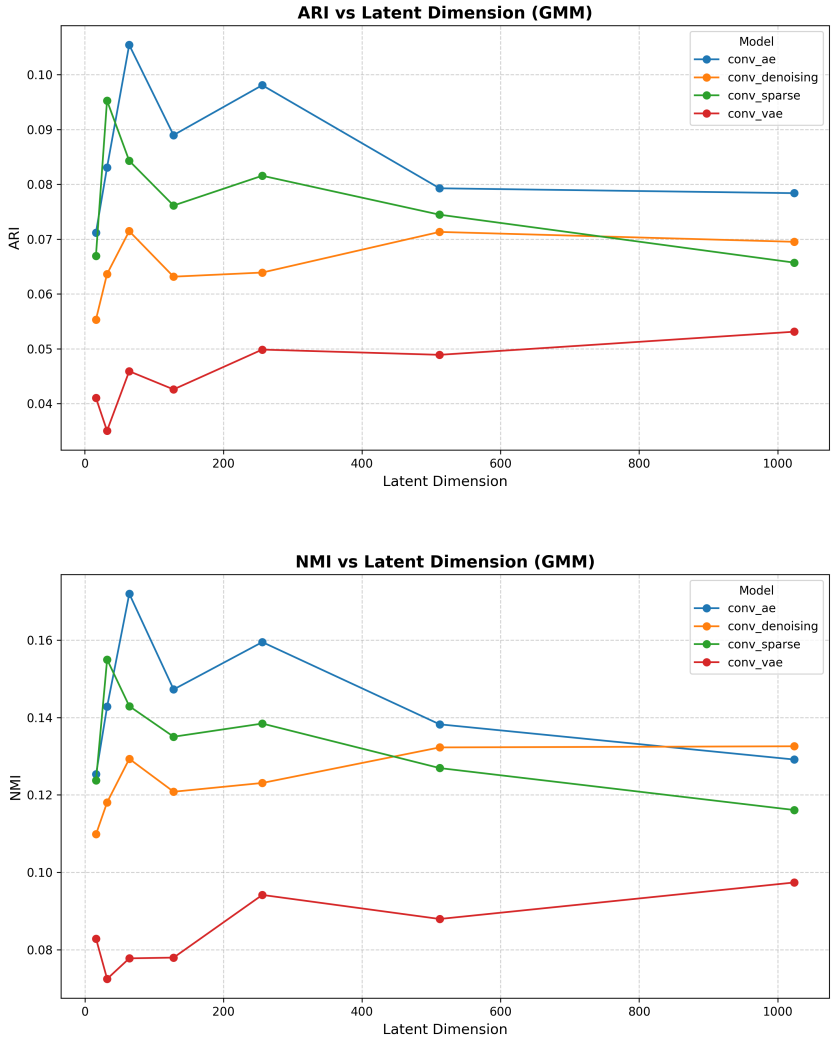


Figure 10: Unsupervised clustering scores (ARI, NMI) across latent dimensions for GMM. Scores are low overall, but GMM recovers modestly higher agreement (NMI up to ≈ 0.17) for some model/dimension pairs.

4.5.2 Overall Assessment

- Deterministic models (`conv_ae`, `conv_sparse`, `conv_denoising`) tend to produce more discriminative embeddings than the variational model under our setup. Observed test accuracies range approximately between 0.35 and 0.60 depending on model, latent size and classifier; the best observed configuration is `conv_denoising` + MLP and `conv_denoising` + LR at values that can reach ≈60
- Sparse regularization (`conv_sparse`) yields modest improvements in some configurations (e.g., `conv_sparse` + MLP peak ≈ 0.5158) but does not guarantee perfect separability; classification remains imperfect across settings.
- The denoising objective provides robustness and, in our experiments, leads to the strongest overall downstream classification when paired with a high-capacity classifier (MLP), particularly at larger latent dimensions. At intermediate dimensions there is a small trade-off between compactness and separability.
- The variational model (`conv_vae`) displays lower accuracy at small latent dimensions and improves as d increases (best observed MLP accuracy are observed at higher latent dimensions), which indicates that the VAE's regularization reduces discriminative information unless its latent capacity is increased.

- Unsupervised clustering metrics remain low overall, but not strictly zero. GMM consistently gives the highest cluster agreement (ARI up to ≈ 0.105, NMI up to $\approx$0.172), especially for some deterministic models at intermediate latent sizes (e.g., `conv_ae` at $d = 64$). KMeans shows smaller but non-negligible scores (ARI $\approx$ 0.03–0.06). HDBSCAN typically fails to recover stable clusters under the default hyperparameters. Together, these results indicate a weak clustering signal in the embeddings: class information is present but not arranged into compact, off-the-shelf clusters.

5 Conclusion and Future Work

This work investigated the impact of latent space size and training objectives on convolutional autoencoder representations. We systematically evaluated reconstruction quality, supervised classification performance, and latent-structure metrics across four architectures (standard, sparse, denoising, variational) and five latent dimensions.

Main findings

- **Reconstruction.** Deterministic sparse autoencoders achieved the best reconstruction fidelity (e.g., `conv_sparse` at $d = 256$: MSE ≈ 0.0021, SSIM ≈ 0.89, PSNR ≈ 27.3). The variational model (`conv_vae`) produced markedly blurrier reconstructions at small d (e.g., $d = 16$: MSE ≈ 0.0638, SSIM ≈ 0.13, PSNR ≈ 12.6), improving only with larger latent capacity.
- **Supervised classification.** The best downstream classification was obtained with `conv_denoising` + MLP and LR with (accuracy ≈ 0.6). Nonlinear classifiers (MLP) consistently outperformed linear models and KNN, indicating that discriminative information is encoded in a non-linear way.
- **Unsupervised clustering.** Unsupervised cluster scores remain low overall: ARI and NMI are typically < 0.20. KMeans produced small but nonzero ARI (roughly 0.05–0.11 across many settings) and GMM recovered a weak signal in some cases (NMI up to ≈ 0.17). HDBSCAN returned mostly noise labels under default parameters.
- **Dichotomy.** There is a clear dissociation between supervised separability and unsupervised clusterability: embeddings can contain class information usable by discriminative classifiers while failing to form compact clusters recoverable by standard clustering algorithms.

Practical implications and recommendations

1. **When reconstruction is the priority:** prefer deterministic architectures (especially sparse variants) or increase VAE capacity / reduce KL weight if using VAEs.
2. **When clustering is required:** adopt objectives that explicitly encourage same-class proximity (contrastive/self-supervised methods such as SimCLR/BYOL or clustering-aware objectives such as DeepCluster/DEC).
3. **Clustering pipeline tuning:** perform targeted hyperparameter searches for clustering (KMeans k, GMM covariances and components, HDBSCAN `min_cluster_size`/`min_samples`) and consider applying PCA/UMAP before clustering.
4. **Robust evaluation:** use nested CV or bootstrap confidence intervals for classifier accuracy, run permutation tests / bootstrap CIs for ARI/NMI, and report per-class metrics (precision, recall, F1) alongside global scores.
5. **Diagnosis and visualization:** complement numeric metrics with qualitative inspection of reconstructions and 2D projections (t-SNE/UMAP) for the best/worst model+d pairs to guide targeted interventions.

Bottom line

Deterministic reconstruction objectives (conv_ae, conv_sparse, conv_denoising) provide superior reconstruction quality and generally more discriminative embeddings under the current protocol; however, embeddings are not reliably cluster-friendly with off-the-shelf clustering methods. Improving unsupervised clusterability requires changes to the learning objective, preprocessing, and careful pipeline tuning.

References

HINTON, Geoffrey E; SALAKHUTDINOV, Ruslan R. Reducing the dimensionality of data with neural networks. **Science**, American Association for the Advancement of Science, v. 313, n. 5786, p. 504–507, 2006.

AN, Jinwon; CHO, Sungzoon. **Variational autoencoder based anomaly detection using reconstruction probability**. [S.l.], 2015. Technical report.

KINGMA, Diederik P; WELLING, Max. Auto-encoding variational Bayes. **arXiv preprint arXiv:1312.6114**, 2013.

ERHAN, Dumitru et al. Why does unsupervised pre-training help deep learning? **Journal of Machine Learning Research**, v. 11, p. 625–660, 2010.

TISHBY, Naftali; ZASLAVSKY, Noga. Deep learning and the information bottleneck principle. In: 2015 IEEE Information Theory Workshop (ITW). [S.l.: s.n.], 2015.

HIGGINS, Irina et al. Beta-VAE: Learning basic visual concepts with a constrained variational framework. In: INTERNATIONAL Conference on Learning Representations (ICLR). [S.l.: s.n.], 2017.

RUMELHART, David E; HINTON, Geoffrey E; WILLIAMS, Ronald J. Learning representations by back-propagating errors. **Nature**, Nature Publishing Group, v. 323, n. 6088, p. 533–536, 1986.

MASCI, Jonathan et al. Stacked convolutional auto-encoders for hierarchical feature extraction. In: INTERNATIONAL Conference on Artificial Neural Networks. [S.l.]: Springer, 2011. v. 6791. (Lecture Notes in Computer Science), p. 52–59.

NG, Andrew. **Sparse autoencoder**. [S.l.: s.n.], 2011. CS294A Lecture Notes, Stanford University.

VINCENT, Pascal et al. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. **Journal of Machine Learning Research**, v. 11, n. 12, p. 3371–3408, 2010.

ZHANG, Richard et al. The unreasonable effectiveness of deep features as a perceptual metric. In: PROCEEDINGS of the IEEE Conference on Computer Vision and Pattern Recognition. [S.l.: s.n.], 2018. P. 586–595.

BENGIO, Yoshua; COURVILLE, Aaron; VINCENT, Pascal. Representation learning: A review and new perspectives. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, IEEE, v. 35, n. 8, p. 1798–1828, 2013.

LOCATELLO, Francesco et al. Challenging common assumptions in the unsupervised learning of disentangled representations. In: INTERNATIONAL Conference on Machine Learning. [S.l.: s.n.], 2019. P. 4114–4124.

BALLE, Johannes; LAPARRA, Valero; SIMONCELLI, Eero P. End-to-end optimized image compression. In: INTERNATIONAL Conference on Learning Representations (ICLR). [S.l.: s.n.], 2017.

KRIZHEVSKY, Alex. **Learning multiple layers of features from tiny images**. [S.l.], 2009.

WANG, Zhou; BOVIK, Alan C et al. Image quality assessment: From error visibility to structural similarity. **IEEE Transactions on Image Processing**, IEEE, v. 13, n. 4, p. 600–612, 2004.

WANG, Zhou; BOVIK, Alan C. Universal image quality index. **IEEE Signal Processing Letters**, IEEE, v. 9, n. 3, p. 81–84, 2002.

WALD, Luc; RANCHIN, Thierry; MANGOLINI, Michel. Quality of high resolution synthesised images: Is there a simple criterion? In: THIRD IEEE International Conference on Image Processing. [S.l.: s.n.], 2000. v. 1, p. 131–134.