Deep Learning Wet Report

Shirah Hassan - 322694019 and Hillah Hassan - 209583574

April 2025

1 Evaluation

1. **Reasoning** In this project, we experimented with different methods of training autoencoders and evaluating the quality of their learned latent representations. We built and trained six models in total - three on the MNIST dataset and three on the CIFAR-10 dataset - each corresponding to one of the three tasks in the assignment: self-supervised autoencoding, classification-guided encoding, and learning a structured latent space using contrastive learning. For each model, we carefully designed the encoder, decoder (if applicable), and classifier, taking into account the characteristics of the data, the goals of the task, and core principles of deep representation learning.

   For the **self-supervised autoencoding task (1.2.1)**, we trained autoencoders in a classical setup, where the encoder compresses an input image into a latent vector, and the decoder attempts to reconstruct the original image from that latent code. For MNIST, which consists of grayscale handwritten digits of size 28×28, we used a fully connected (MLP) architecture for both the encoder and decoder. The encoder consists of three hidden layers: the first maps the 784-dimensional flattened input to 256 units, followed by a ReLU activation; the second maps 256 to 128 with another ReLU; and the final layer maps 128 to the 128-dimensional latent vector (this last one is considered the output layer, not a hidden layer). The decoder mirrors this structure in reverse: two hidden layers that expand from 128 to 256 and then to 784, ending in a Sigmoid activation to produce pixel outputs in [0,1]. We chose this relatively shallow structure because MNIST is a simple dataset where deeper networks are unnecessary and may overfit.

   For CIFAR-10**,** which contains more complex 32×32 RGB images, we adopted a CNN-based architecture. The encoder has **four hidden layers**: it uses a stack of convolutional layers with increasing channel depth (starting at 3 input channels), each followed by batch normalization and ReLU activations. After flattening, the final projection maps the feature maps to a 128-dimensional latent vector. The decoder reverses this with

three deconvolutional (ConvTranspose2d) hidden layers, gradually increasing spatial resolution back to 32×32. We selected this design because CNNs are well-suited to capture local spatial features like edges and textures, which are vital for reconstructing CIFAR images accurately.

In the **classification-guided encoding task (1.2.2)**, our goal was to learn a latent space optimized for classification accuracy rather than reconstruction. For this task, we reused the encoder architectures from the self-supervised task. For MNIST, the encoder still had **three hidden layers**, and we connected its output to a two-layer MLP classifier. This classifier has **one hidden layer**, which maps from 128 to 64 units with a ReLU, followed by a final layer projecting to 10 classes. For CIFAR-10, we used the same CNN encoder as before and the same two-layer classifier. These models were trained end-to-end using a cross-entropy loss function, allowing the encoder to learn features specifically tailored for class separation.

In the **structured latent space task (1.2.3)**, we aimed to improve the organization and separability of the latent space by introducing contrastive learning, rather than directly optimizing for reconstruction or classification. For this, we used the same encoder architectures as before, without a decoder. In contrastive training, each input image was augmented in two different ways to create a pair of similar (positive) samples. Dissimilar (negative) samples were drawn from the rest of the batch. We used an **NT-Xent (Normalized Temperature-Scaled Cross Entropy) loss** function to train the encoder such that similar images (under augmentation) had nearby representations in latent space, while dissimilar ones were farther apart. For MNIST, this was again a **three-layer MLP encoder**, while for CIFAR-10 we used the **four-layer convolutional encoder**. After training, we froze the encoders and trained a two-layer classifier on top to evaluate how well the contrastive training improved separability.

Across all models and datasets, we fixed the latent vector size to 128 to ensure consistent comparison. Hyperparameter tuning was done using a validation set. We trained all models using the Adam optimizer with a learning rate of 0.001 and a batch size of 128. Training spanned 20 to 30 epochs depending on convergence behavior. We also used dropout layers in classifiers to reduce overfitting and applied early stopping when necessary. For autoencoder models, reconstruction error was measured using **mean absolute error (MAE)**, while classification performance was evaluated using accuracy.

In summary, we chose relatively shallow architectures (2–4 hidden layers depending on the model) to balance model complexity with performance, especially given the small size and resolution of the datasets. For MNIST, we relied on simple MLPs, while for CIFAR-10 we used convolutional networks to better capture spatial information. Our contrastive learning method successfully enhanced the structure of the latent space, as shown by improved downstream classification and visualization, and each design

decision was made with clarity on how the model structure aligns with the learning objective.

2. **Quantitative Results (MNIST)**

   To evaluate the quality of the latent representations learned by each model, we measured classification accuracy on the training, validation, and test sets across all three training methods: self-supervised autoencoding, classification-guided encoding, and contrastive learning. For the self-supervised autoencoder (1.2.1.1), we also computed mean absolute error (MAE) between input and reconstructed images to assess how well the autoencoder captures important image features.

   In the **self-supervised setting (1.2.1.1)**, the autoencoder achieved a **training MAE of 0.011433**, a **validation MAE of 0.011733**, and a **test MAE of 0.011579**. These low reconstruction errors indicate that the autoencoder successfully learned to compress and decompress MNIST images with minimal information loss. We then trained a separate classifier on top of the frozen encoder (1.2.1.2), which achieved a **training accuracy of 98.80%**, **validation accuracy of 97.52%**, and **test accuracy of 97.64%**. This result shows that even though the encoder was not explicitly trained for classification, its representations still captured semantically useful features.

   In the **classification-guided approach (1.2.2)**, where the encoder and classifier were trained end-to-end using supervised learning, the model achieved a **training accuracy of 99.53%**, a **validation accuracy of 98.48%**, and a **test accuracy of 98.61%**. This slightly outperformed the self-supervised approach, which is expected since the encoder was directly optimized for label prediction rather than reconstruction. These results serve as an upper bound for encoder performance in this setting.

   Finally, the **contrastive learning model (1.2.3)** achieved strong results as well. After training the encoder using NT-Xent loss with data augmentations, we froze the encoder and trained a classifier on top. This model achieved a **training accuracy of 99.51%**, **validation accuracy of 98.42%**, and **test accuracy of 98.54%**. While it did not surpass the classification-guided encoder, it performed better than the self-supervised encoder and provided a much more structured latent space, as confirmed by our t-SNE visualizations.

Method	Training Accuracy	Validation Accuracy	Test Accuracy	Training MAE
Self-Supervised Autoencoder (1.2.1.1)	-	-	-	0.011433
Classifier on Frozen Encoder (1.2.1.2)	98.80%	97.52%	97.64%	-
Classification-Guided Autoencoder (1.2.2)	99.53%	98.48%	98.61%	-
Contrastive Learning Autoencoder (1.2.3)	99.51%	98.42%	98.54%	-

Table 1: Quantitative Results for MNIST Autoencoders. MAE is reported for the self-supervised autoencoder.

Overall, contrastive and classification-guided encoders yield the most sep-

arable representations.

Quantitative Results – CIFAR-10

We evaluated four encoder models on CIFAR-10 across three paradigms:two self-supervised autoencoders, a classification-guided encoder, and a contrastive learning encoder. For each, we report classification accuracy on the training, validation, and test sets. For the autoencoder-based encoders, we additionally report mean reconstruction error, using Mean Absolute Error (MAE).

The first two encoders were trained as self-supervised autoencoders—one without skip connections and one with skip connections. After training, a linear classifier was trained on top of the frozen encoders. The non-skip model achieved a training accuracy of **61.76%**, validation accuracy of **60.91%**, and test accuracy of **61.20%**, with a reconstruction MAE of **0.0615**. The skip-connected model improved on all fronts, yielding **67.91%** training accuracy, **65.38%** validation accuracy, and **64.98%** test accuracy, with an MAE of **0.0597**. These results suggest that architectural improvements like skip connections enhance both representation quality and reconstruction fidelity.

The classification-guided encoder was trained end-to-end on the classification task. It yielded the strongest results overall, reaching **99.63%** accuracy on the training set, **89.12%** on the validation set, and **88.66%** on the test set. As expected, this supervised model outperformed all other encoders in downstream classification accuracy. Since it lacked a decoder, we do not report MAE for this model.

The final encoder, based on contrastive learning (Section 1.2.3), was trained using a hybrid loss combining NT-Xent contrastive loss with a denoising autoencoding objective. The resulting encoder was evaluated by training a linear classifier on its frozen embeddings. This model achieved **73.06%** training accuracy, **71.36%** validation accuracy, and **71.36%** test accuracy. Its reconstruction performance yielded a test MAE of **0.1644**, considerably higher than the pure autoencoders, consistent with the fact that contrastive learning emphasizes discriminative feature learning rather than pixel-level reconstruction.

Table 2: Downstream classification accuracy (Train / Val / Test)

Model	**Train Acc**	**Val Acc**	**Test Acc**
Autoencoder (no skip)	61.76%	60.91%	61.20%
Autoencoder (skip)	67.91%	65.38%	64.98%
Supervised Encoder	99.63%	89.12%	88.66%
Contrastive Encoder	73.06%	71.36%	71.36%

Table 3: Mean Absolute Error (MAE) on Test Set

Model	**Test MAE**
Autoencoder (no skip)	0.0615
Autoencoder (skip)	0.0597
Contrastive Encoder	0.1644

3. **Qualitative Results**

Figure 1: 5 reconstructed images from MNIST

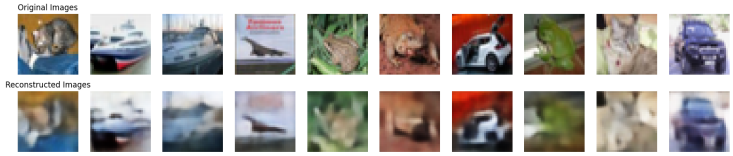


Figure 2: 10 reconstructed images from CIFAR

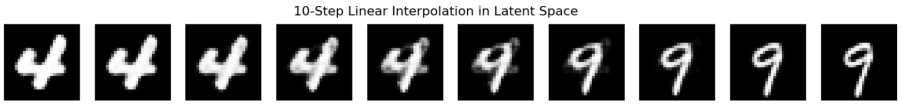


Figure 3: Explanation below.

4. To see how well our self-supervised autoencoder captures the structure of the data, we ran a 10-step linear interpolation in the latent space between two MNIST test images: a **4** and a **9**. Both images were encoded using the frozen encoder from section 1.2.1, and we decoded the interpolated latent vectors using the pre-trained decoder.

   The interpolated images show a smooth, visually coherent transition 4 to 9. At the start of the sequence, the image clearly looks like a 4, but as we move through the latent space, we see features like the closed loop and more vertical stroke start to appear which is more typical of a 9.

While some of the intermediate images don't perfectly resemble valid digits, they still maintain a handwriting-like structure. We hypothesize this means that the latent space is relatively smooth and semantically meaningful, even though the model wasn't explicitly trained to interpolate digits. In other words, the fact that we get this kind of smooth transition shows that the self-supervised encoder is embedding digits in a way that places similar classes or visual features close to each other.

The decoder does a great job of mapping the continuous latent codes back into plausible images. This means that even the interpolated points between two real encodings give us reasonable outputs. This is a strong indication that our autoencoder has learned useful, interpretable features, even without labels during training.

5. **t-SNE Analysis: Our CIFAR-10 and MNIST Autoencoders**

   To understand the effect of different training strategies on learned representations, we visualized t-SNE projections of both the **latent space** and the **image space** for each of the six autoencoder variants trained on CIFAR-10 and MNIST:

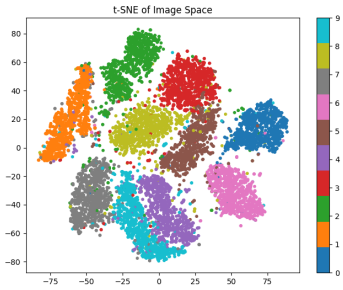
   - a **self-supervised autoencoder**,
   - a **classification-guided autoencoder**, and
   - a **contrastive (self-supervised) autoencoder**.

   To evaluate each model, we applied t-SNE to both the latent representations and the original image space, which allowed us to assess how effectively class information was encoded.

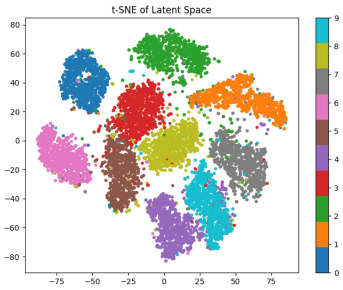
t-SNE Analysis of Latent and Signal-Domain Representations MNIST

Differences Between Latent Spaces from Self-Supervised and Classification-Guided Training We looked at the latent space visualizations from three different training strategies: (1) self-supervised autoencoding (section 1.2.1), (2) classification-guided training (section 1.2.2), and (3) contrastive training (section 1.2.3). We used t-SNE to project the latent spaces into 2D and then analyzed how the resulting clusters aligned with the true class labels.

Let's start with presenting the results of the self-supervised autoencoder.



(a) Image Space (Self-Supervised Autoencoder)

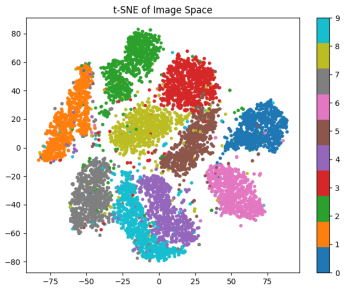


(b) Latent Space (Self-Supervised Autoencoder)

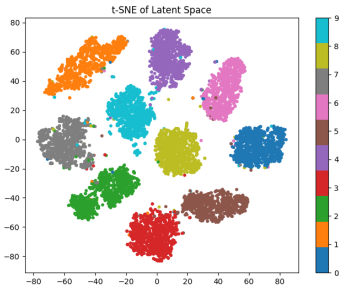
Figure 4: t-SNE visualizations for the Self-Supervised Autoencoder (1.2.1).

The latent space of the **self-supervised autoencoder** exhibits poor class separation. As seen in the image space in Figure 4, some digits appear similar to others, leading to cases where red dots, for example, appear outside the red cluster. Overall, the different classes are somewhat clustered in image space, though there is overlap. Then, on the right, in the latent space, we still observe clusters, but they are far from tight compared to other models. In fact, it's hard to argue that the latent space clusters are any more organized than the image space clusters. This suggests that the autoencoder has not learned particularly meaningful features. We hypothesize that the model has prioritized input reconstruction over encoding semantically meaningful information. Since the training objective focuses on reconstructing the input , the latent representation captures low-level features essential for reconstruction, rather than high-level semantic distinctions.

Let's compare this to the results of the classification-guided autoencoder.



(a) Image Space (Classification-Guided Encoder)

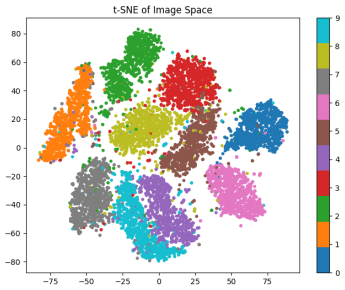


(b) Latent Space (Classification-Guided Encoder)

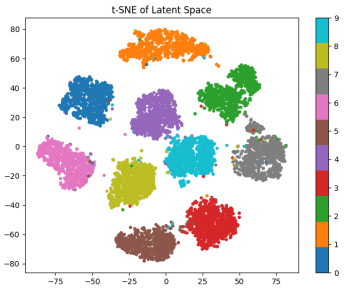
Figure 5: t-SNE visualizations for the Classification-Guided Encoder (1.2.2).

The latent space learned by the **classification-guided autoencoder** shows the clearest class separation of the three. Each class forms a compact, well-defined cluster with clear boundaries between them. This makes sense: since the encoder is trained with classification loss, it's directly pushed to create embeddings that are linearly separable. The classification objective forces embeddings from the same class to come together while keeping different classes apart, which leads to a latent space structure that's optimized for classification tasks down the line.

Let's compare this to the results of the self-supervised contrastive autoencoder.



(a) Image Space (Contrastive Autoencoder)



(b) Latent Space (Contrastive Autoencoder)

Figure 6: t-SNE visualizations for the Contrastive Autoencoder (1.2.3).

For the contrastive autoencoder (Section 1.2.3), the t-SNE plot of the latent space (Figure 5) shows an arguably moderate improvement in class

separation compared to the classification guided autoencoder. The clusters are farther apart, and while there's still some overlap — especially between similar digits like 4s and 9s — the overall structure is noticeably better. The contrastive model has managed to bring similar instances closer together in the latent space.

These visual trends align with the measured downstream classification accuracies, where the contrastive model outperformed the rest.

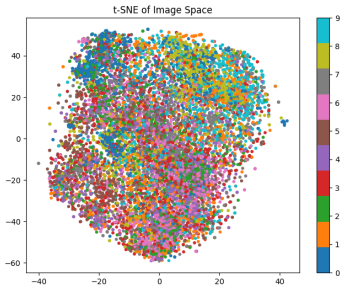
t-SNE Analysis of Latent and Signal-Domain Representations CIFAR-10

To understand the effect of different training strategies on learned representations, once again, we visualized t-SNE projections of both the **latent space** and the **image space** for each of the six autoencoder variants trained on CIFAR-10:

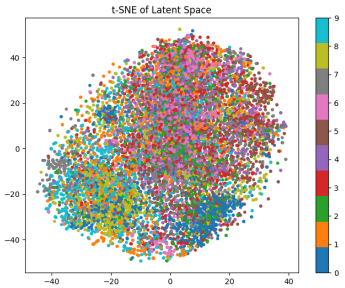
- a **self-supervised denoising autoencoder**,
- a **classification-guided autoencoder**, and
- a **contrastive (self-supervised) autoencoder**.

Each model was evaluated by applying t-SNE on both its latent representations and on the original image space, allowing us to observe how well class information was encoded in each case.

Differences Between Latent Spaces from Self-Supervised and Classification-Guided Training We'll begin by presenting the results of the self-supervised autoencoder.



(a) Image Space (Self-Supervised Denoising AE)

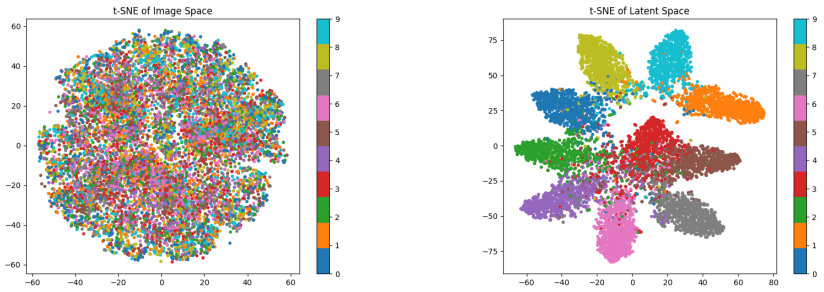


(b) Latent Space (Self-Supervised Denoising AE)

Figure 7: t-SNE visualizations for the self-supervised denoising autoencoder.

The latent space of the **self-supervised (denoising) autoencoder** shows poor class separation, with all classes appearing entangled and overlapping in a big blob in both the signal space and latent space. Given that we're working with CIFAR-10, a more complex dataset than MNIST, it's not surprising that the image and latent space look like an even bigger mess here. This suggests that the model has focused more on reconstructing the input signals rather than learning semantically meaningful features. Since the training objective is to reconstruct the input (possibly with added noise), the latent representation captures low-level features necessary for reconstruction, but not necessarily high-level distinctions that would separate the different classes.

Let's compare to the classification guided autoencoder.



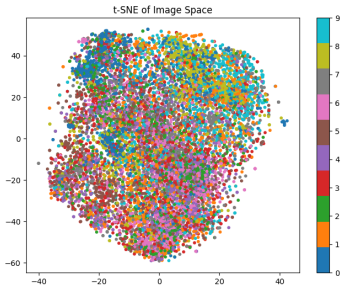
(a) Image Space (Classification-Guided AE)

(b) Latent Space (Classification-Guided AE)

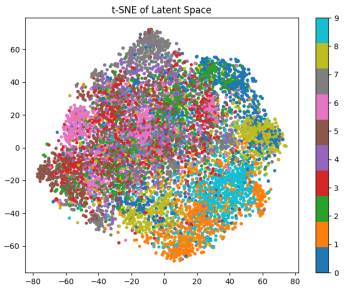
Figure 8: t-SNE visualizations for the classification-guided autoencoder.

The latent space learned by the **classification-guided autoencoder** is the most class-separable of the three. Each class forms a tight, distinct cluster, with clear separation between them. This result is expected, as we discussed earlier, but it's even more impressive here compared to MNIST. In the image space, we see one large, messy blob with significant overlap, whereas in the latent space, the clusters are much tighter and well-defined!

Now we'll present the results of the contrastive model.



(a) Image Space (Contrastive AE)



(b) Latent Space (Contrastive AE)

Figure 9: t-SNE visualizations for the contrastive autoencoder.

Our contrastive autoencoder model (section 1.2.3) combines denoising reconstruction loss with a contrastive objective that pushes similar inputs (like augmentations of the same image) to be close together in the latent space. This approach gives better class separation than the self-supervised autoencoder, but it still doesn't quite match up to the classification-guided model.

Looking at the t-SNE visualization of the contrastive latent space, we see some moderate clustering by class — it's an improvement over the tangled mess of the self-supervised model, but not nearly as clean and well-defined as the classification-guided one. Some clusters are only partially formed, with overlap between visually similar classes.

This shows that contrastive learning helps the model get closer to having separable clusters, even without class labels, by teaching it to distinguish between similar instances. However, without class-level supervision, the model can't quite align its latent space with the actual semantic categories. So while contrastive learning definitely improves the quality of the learned representations, it doesn't quite close the gap with classification-guided training.

These visual trends align with the measured downstream classification accuracies, where classification-guided models outperform contrastive models, which in turn outperform self-supervised denoising models.

We'll now explicitly address **Signal-Domain vs. Latent Representations.**

We also took a look at the t-SNE visualizations of both the original image space (the raw pixel data, or "signal-domain") and the learned latent representations. For all models, the image space projections show a pretty messy, overlapping cloud of points with no clear class separation. This makes sense because raw pixel values are noisy and don't have any inherent structure — there's a lot of variation within classes and a lot of similarity between different classes in terms of raw pixel values.

On the other hand, the latent space projections, especially for the classification-guided and contrastive models, show much clearer structure. The encoder has learned to map the raw input images into a more organized and compact space,

where intra-class variance is reduced and classes are more distinct from each other. This really highlights the role of the encoder: it's transforming high-dimensional, noisy input into more useful, task-specific representations that make it easier to separate and classify the data.