Autoencoders Part B

Outline

- What are autoencoders?
- Brief history of encoding/decoding.
- Inside autoencoders.
- **Convolutional autoencoders.**
- **Regularization of autoencoders.**
- **Applications**
  - **Denoising**
  - **Blending**

Autoencoder with fully connected layers

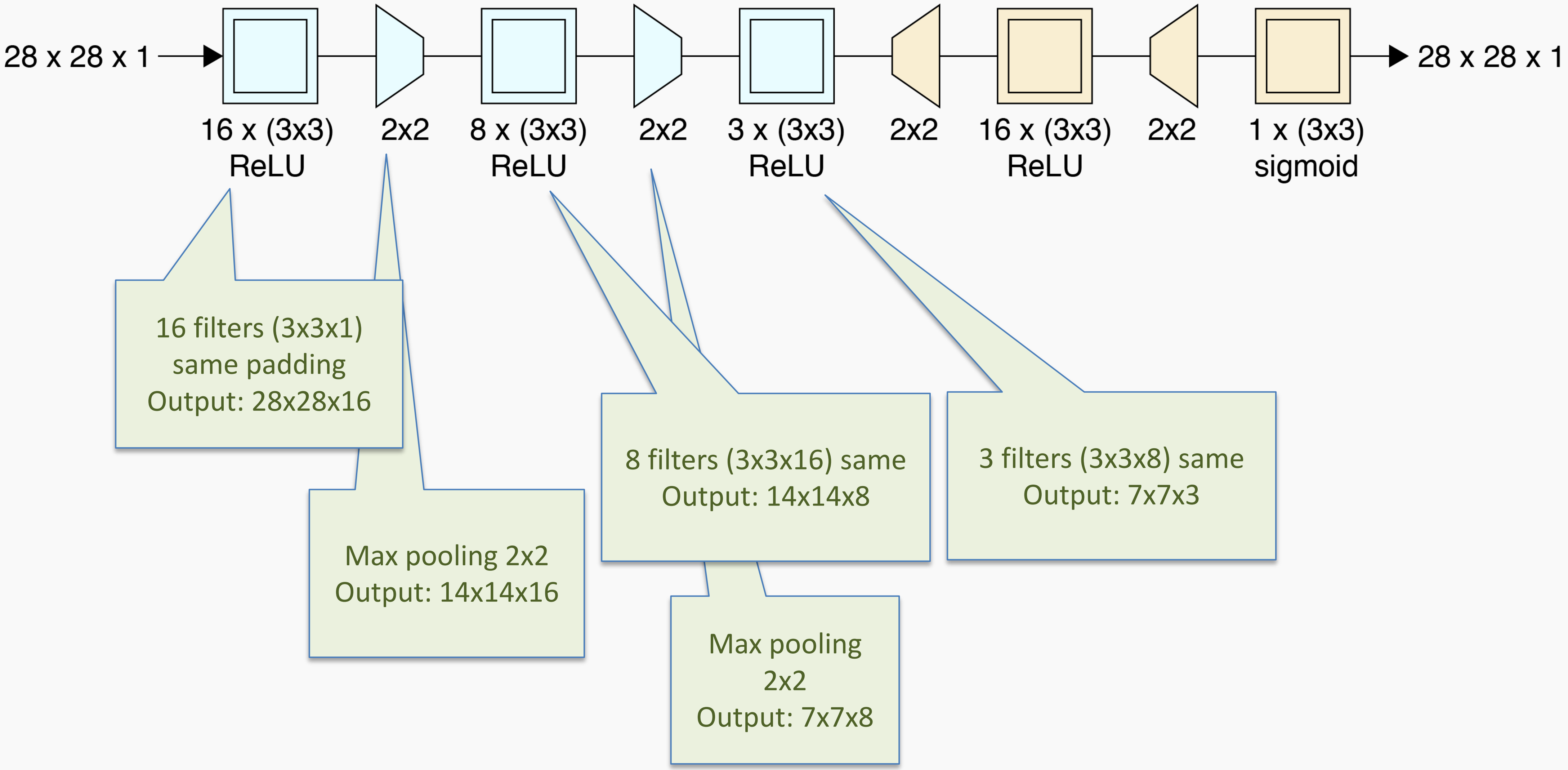
Check again if the deep AE trained on MNIST works with "Pavlos" image?

512 neurons → 256 neurons → 20 neurons → 256 neurons → 512neurons → 784 neurons

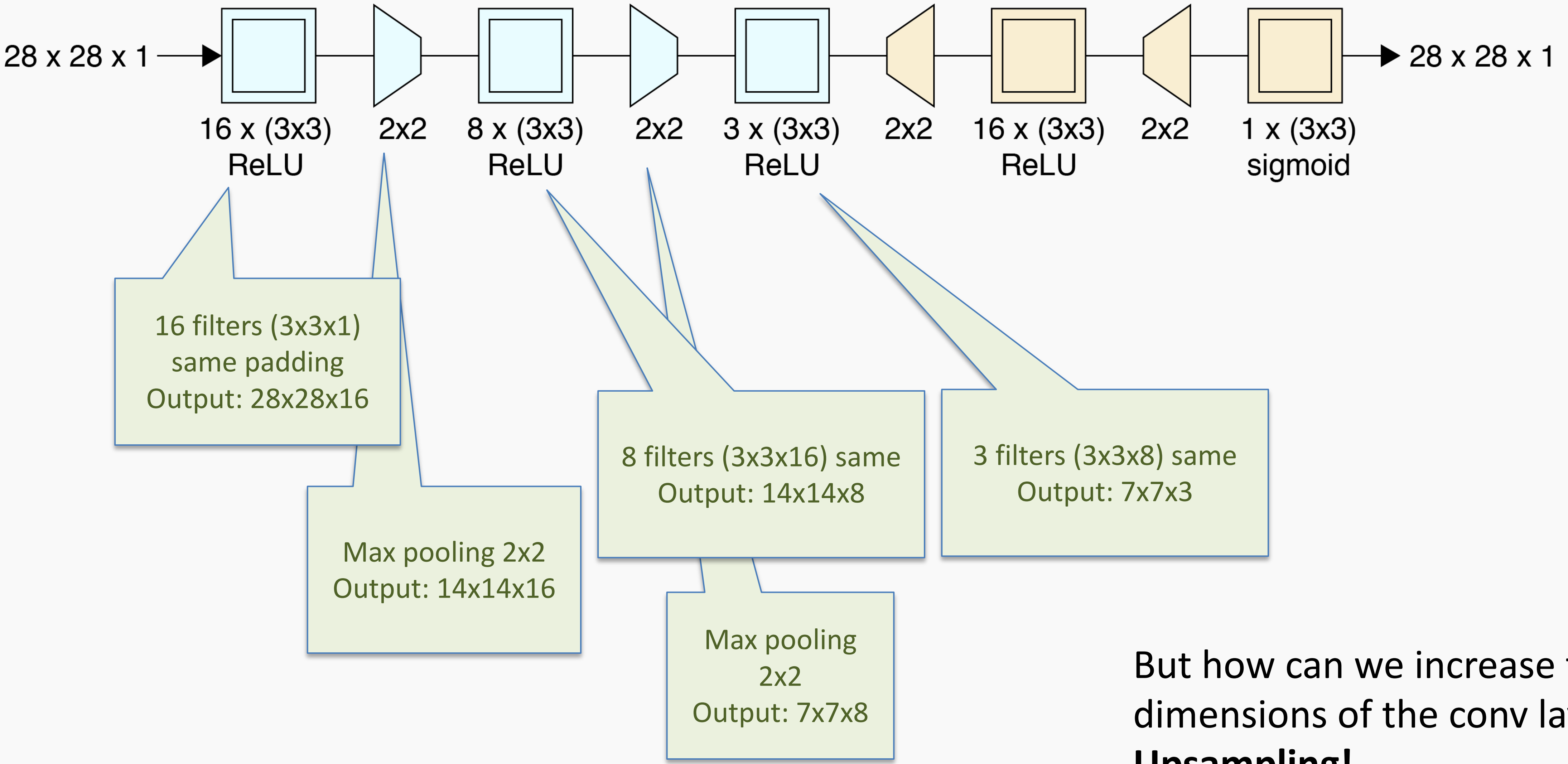
NO

Since we are dealing with images, it is best to use CNNs

Convolutional Autoencoders

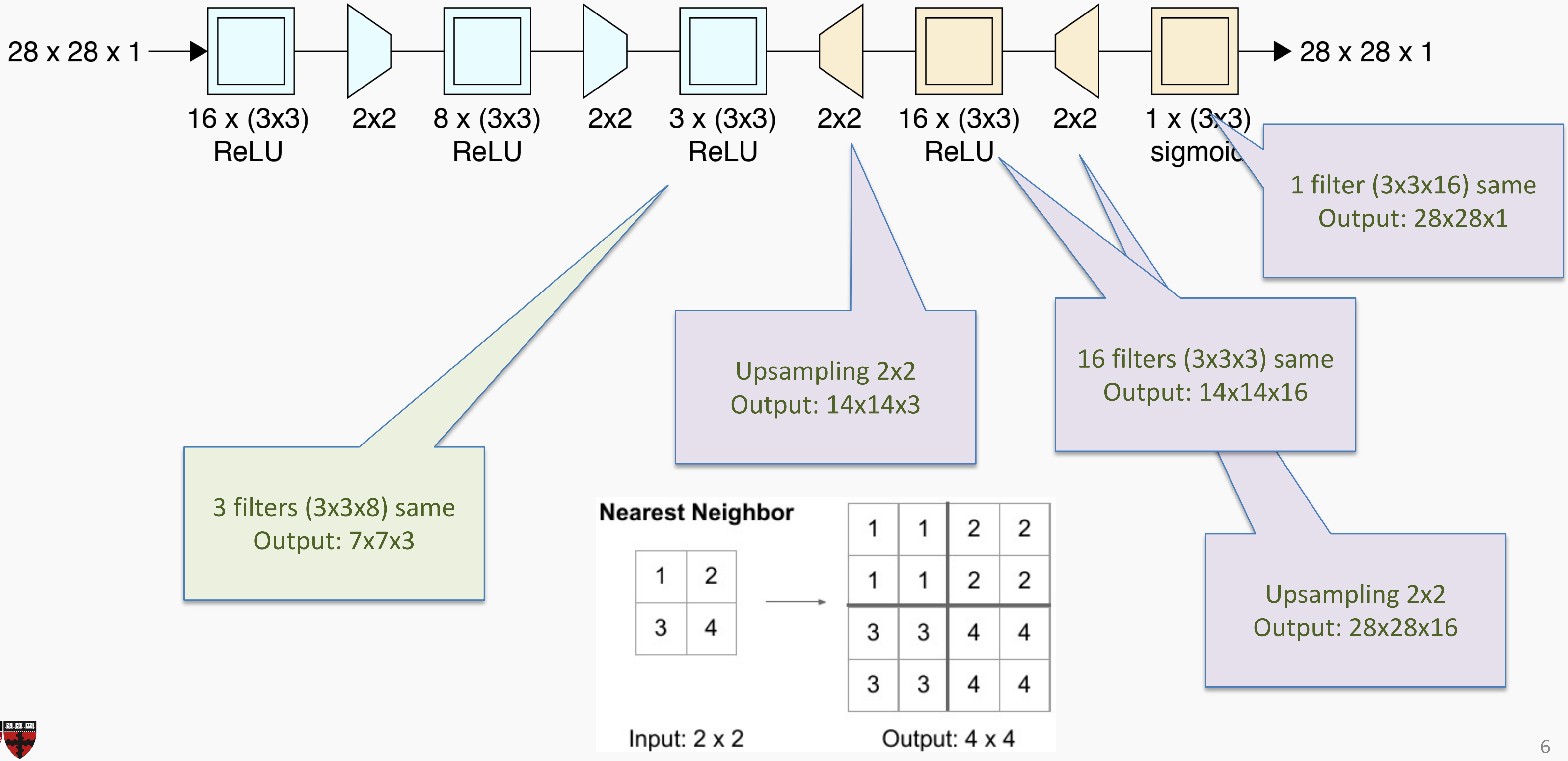


Convolutional Autoencoders

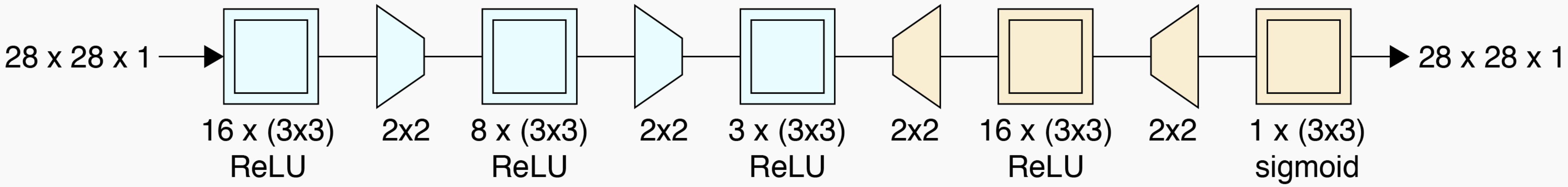


But how can we increase the dimensions of the conv layers? **Upsampling!**

Convolutional Autoencoders

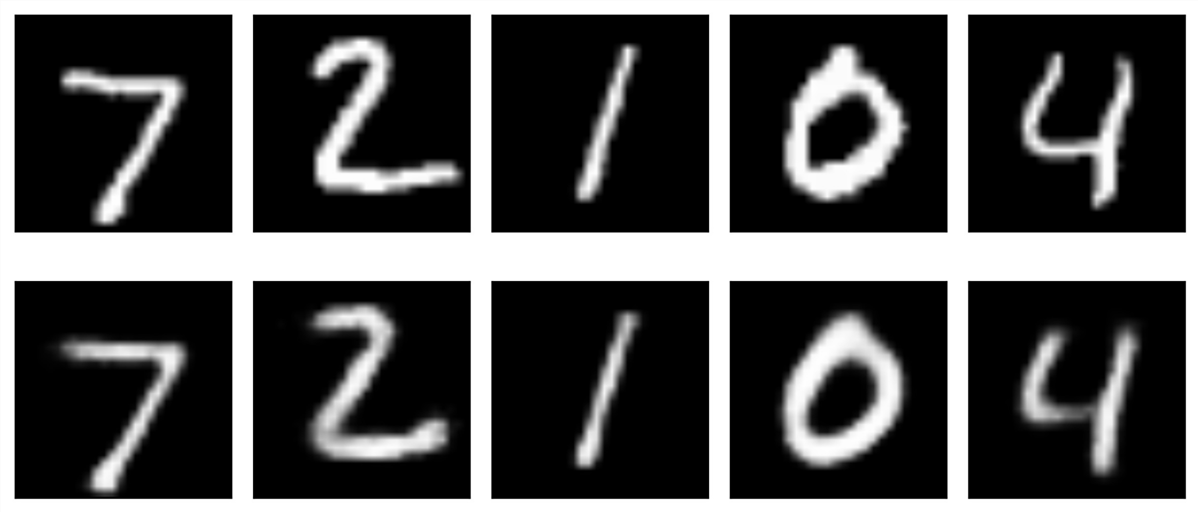


Convolutional Autoencoders

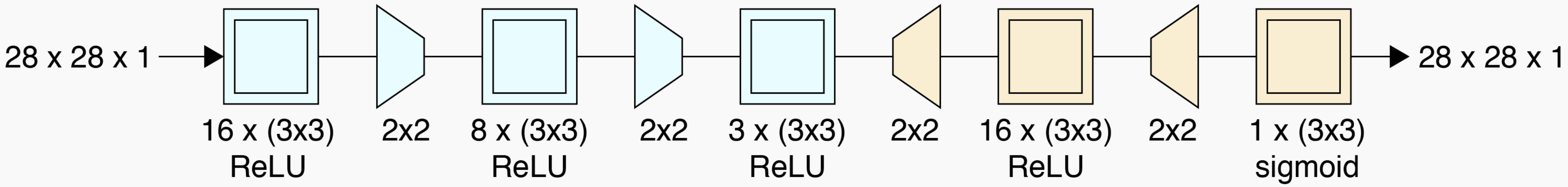


Original Images

Reconstructed Images
with **DeepFCN AE**

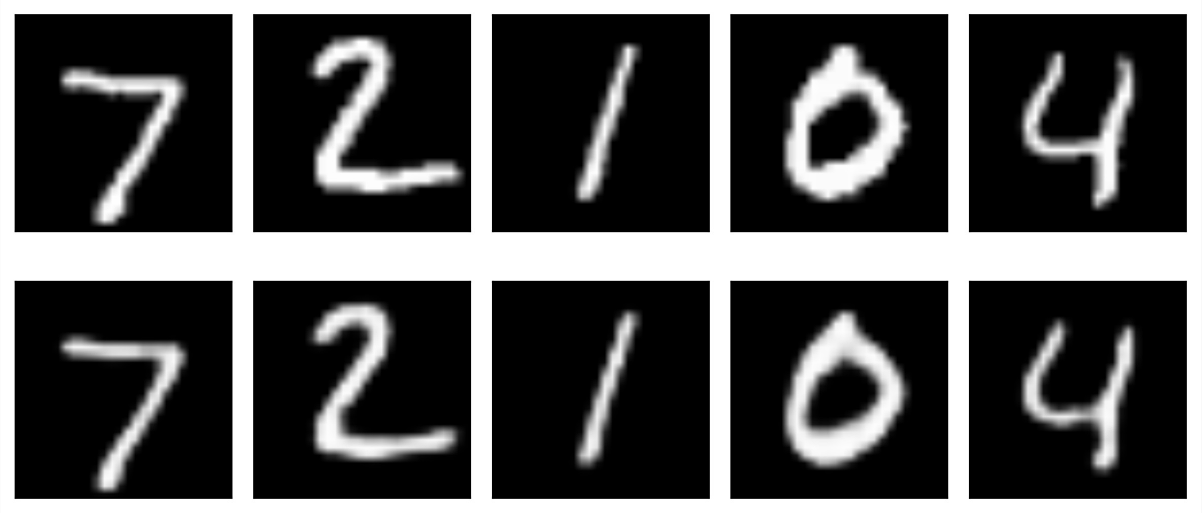


Convolutional Autoencoders



Original Images

Reconstructed Images
with **Conv AE**



Regularization of Autoencoders

- Sparse autoencoders
- Contractive autoencoders
- Denoising autoencoders

Sparse Autoencoders

This trade-off requires the model to learn only the variations in the data required to reconstruct the input. Avoid holding on to redundancies within the input.

Question: How to achieve this?

Add a second loss term that encourages low-dimensional latent space (**sparsity penalty**).

$$\mathcal{L}(x, \hat{x}) + \Omega(z)$$

Regularization on the output of encoder (latent space), not on network parameters.

The first term encourages our model to be sensitive to the inputs (reconstruction loss) and a second term discourages memorization/overfitting (regularization).

Sparse Autoencoders

Apply an L1 regularization on the bottleneck activated neurons (latent space)

$$\mathcal{L}(x, \hat{x}) + \lambda \sum_i |z_i|$$

- We ask the AE to have the lowest possible dimensional latent space that is sufficient to reconstruct the input data.
- Limit the network's capacity to memorize the input data without limiting the networks capability to extract features from the data.
- Individual regions of the AE are selectively activated depending on the input. Each region takes care of a specific attribute of the input data.

Contractive Autoencoders

Intuitively, we would expect that for very similar inputs, the learned encoding would also be very similar.

In other words, we want the latent space to not change a lot when the input data slightly changes.

How can we assist the AE to do that?

Derivatives! Remember the saliency maps

Apply an L2 regularization on the derivatives of the latent variables

$$\mathcal{L}(x, \hat{x}) + \lambda \sum_i ||\nabla_x z_i||^2$$

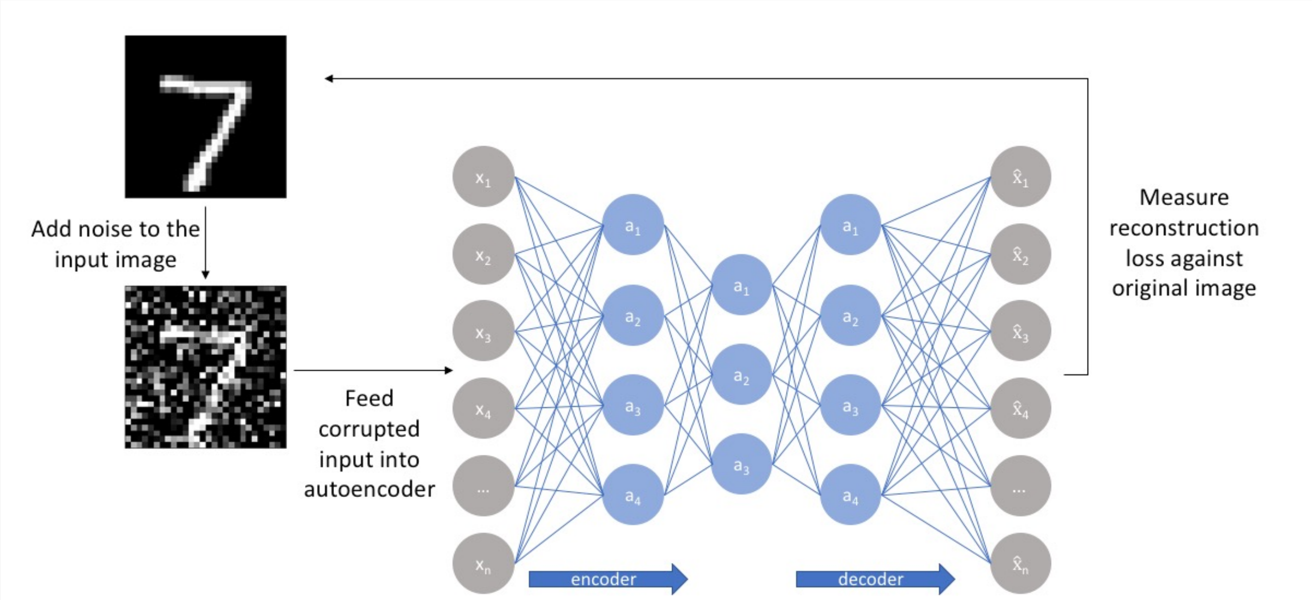
Denoising Autoencoders

Make the AE predictions non-sensitive to noise input images

We feed an AE with noisy data point (input) and and train to predict the original

For each epoch:

1. Add noise to the input image
2. Forward the noisy image through the AE
3. Calculate the reconstruction loss with the original image



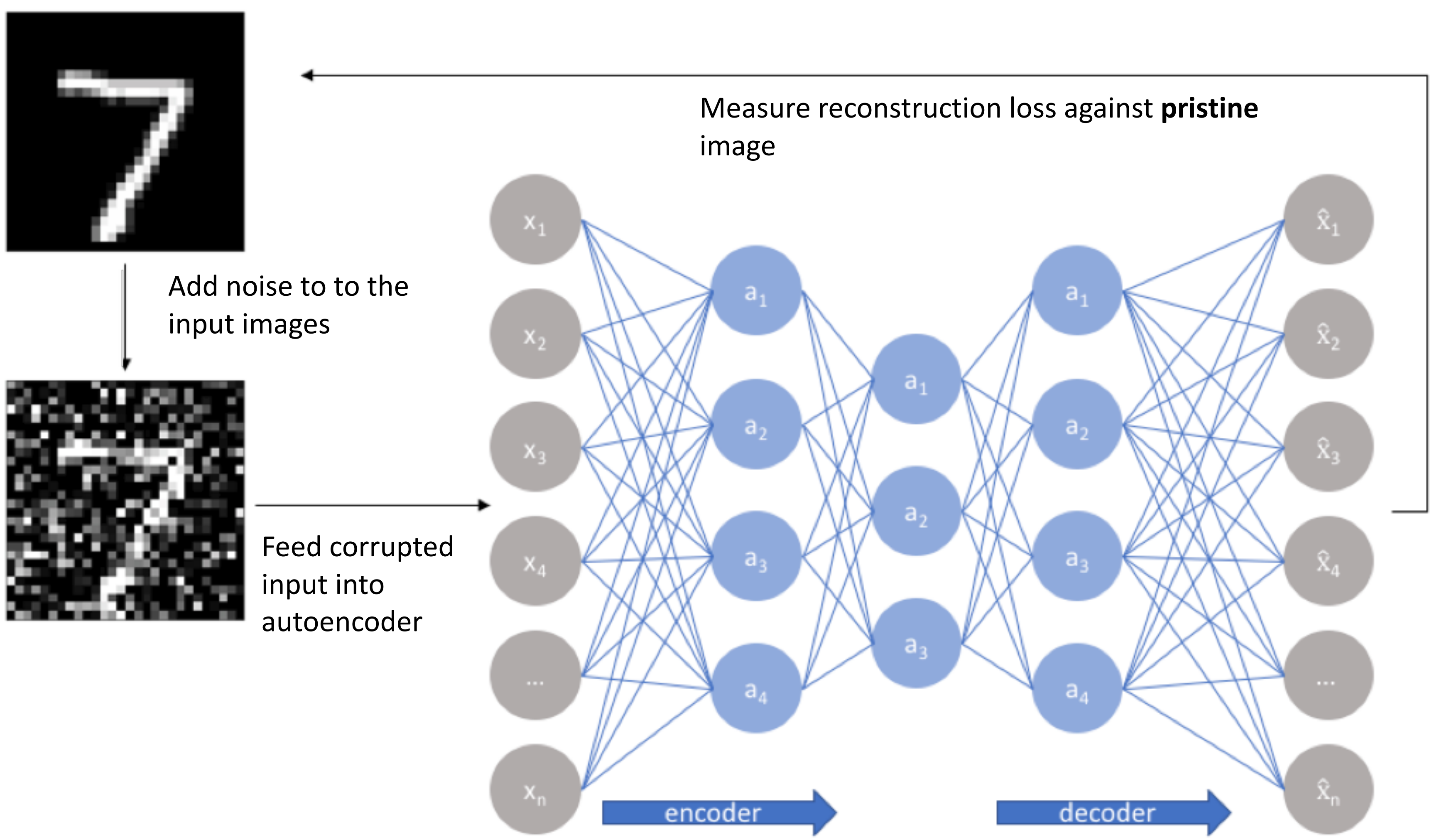
Applications of Autoencoders

- Denoising images
- Blending

Denoising images

A popular use of autoencoders is to remove noise from samples.

Start with a **pristine** image



Blending

We blend inputs to create new data that is similar to the input data, but not exactly the same.

One example is the **content blending** where the content of two pieces of data is directly blended. An example is if we overlay images of a cow and zebra.

Image taken from A. Glassner, Deep Learning, Vol. 2: From Basics to Practice

Content Blending

Content blending on MNIST images

Image taken from A. Glassner, Deep Learning, Vol. 2: From Basics to Practice

Representation blending

Another type of blending is **representation** blending:

In this type of blending, we take advantage of the contextual learning to describe the objects we are interested in.

By engaging in blending in the latent (parameter) space, we create data that blend the essence or the inherent qualities of the objects of interest.

Representation blending (cont)

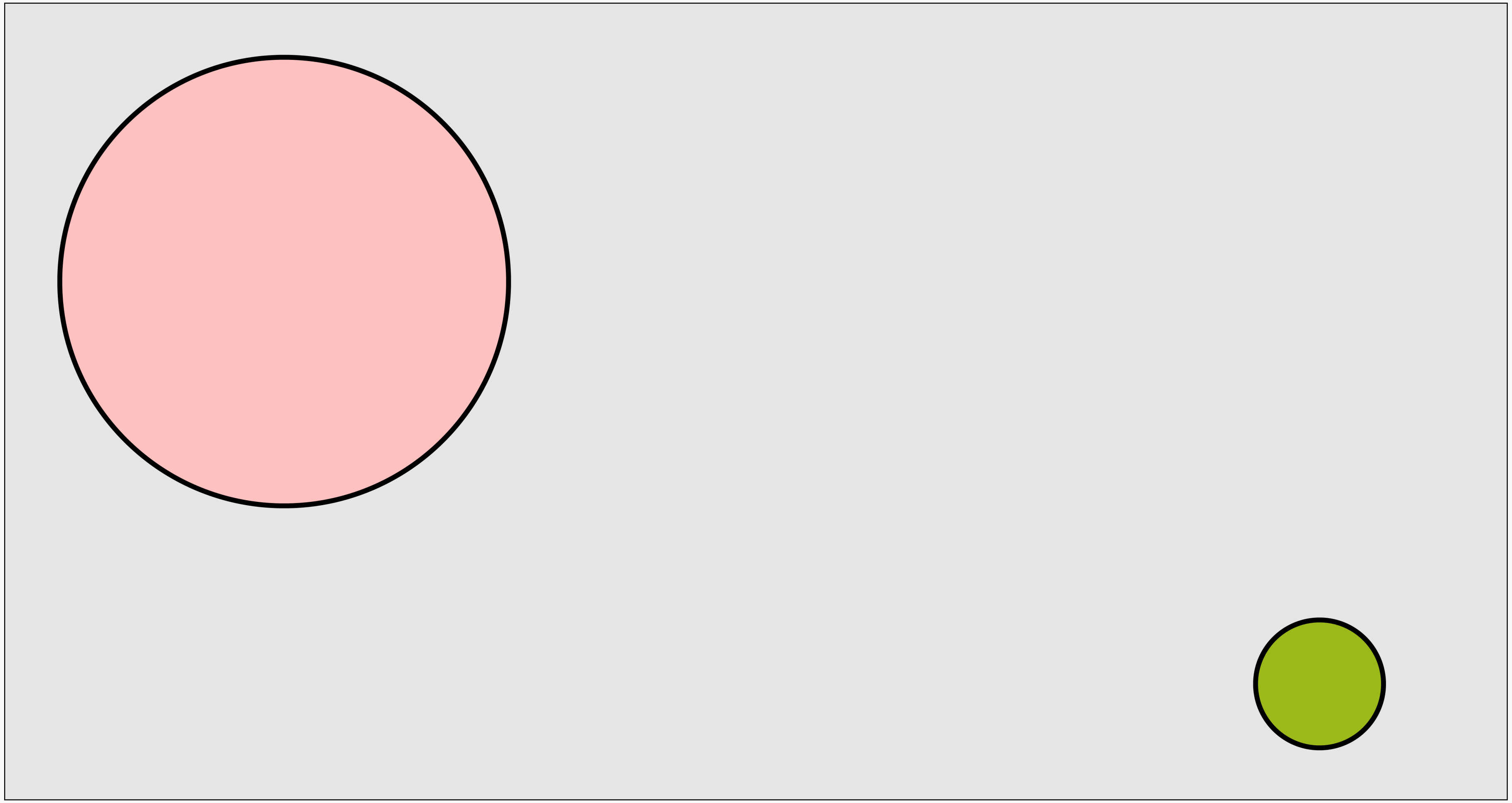


Image taken from A. Glassner, Deep Learning, Vol. 2: From Basics to Practice

Representation blending

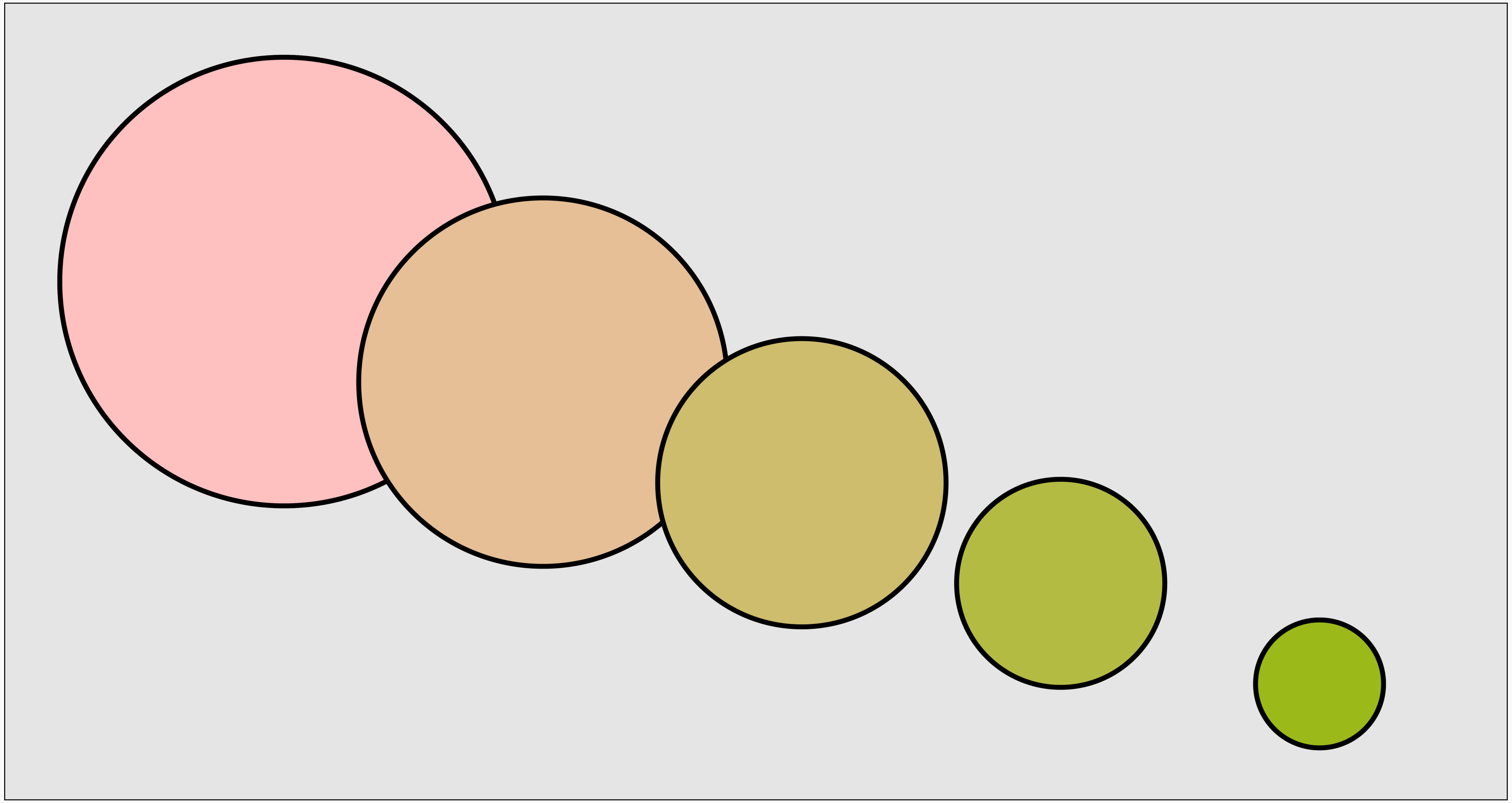
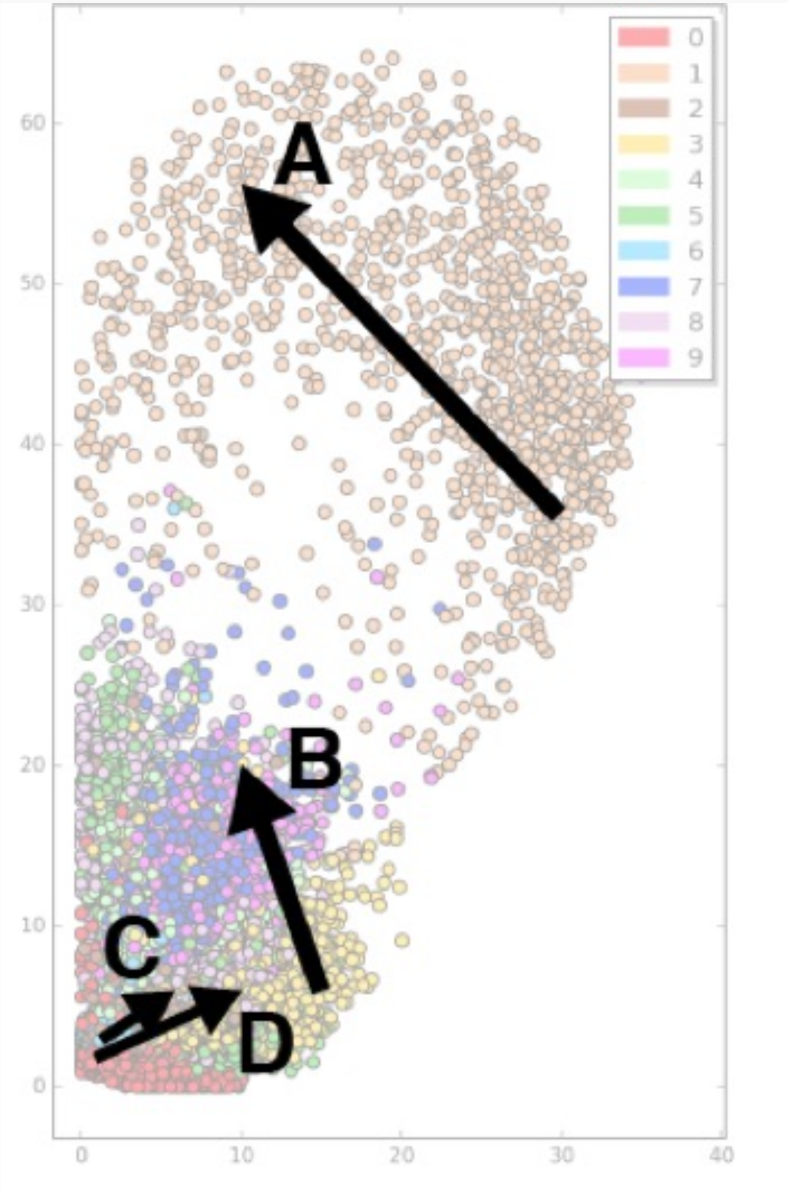


Image taken from A. Glassner, Deep Learning, Vol. 2: From Basics to Practice

Blending Latent Variables



Back to the example of MNIST.

1. We start at the start of the arrows in latent space and then move to end of the arrow in 7 steps.
2. For each value of Z we use the already trained decoder to produce an image.

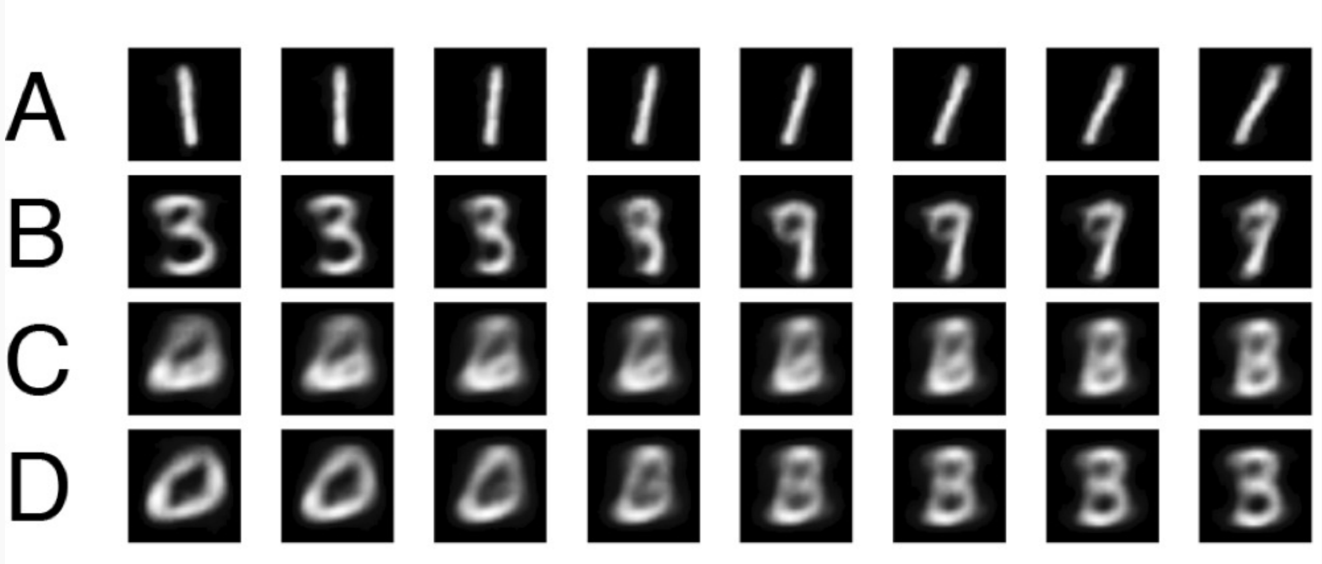
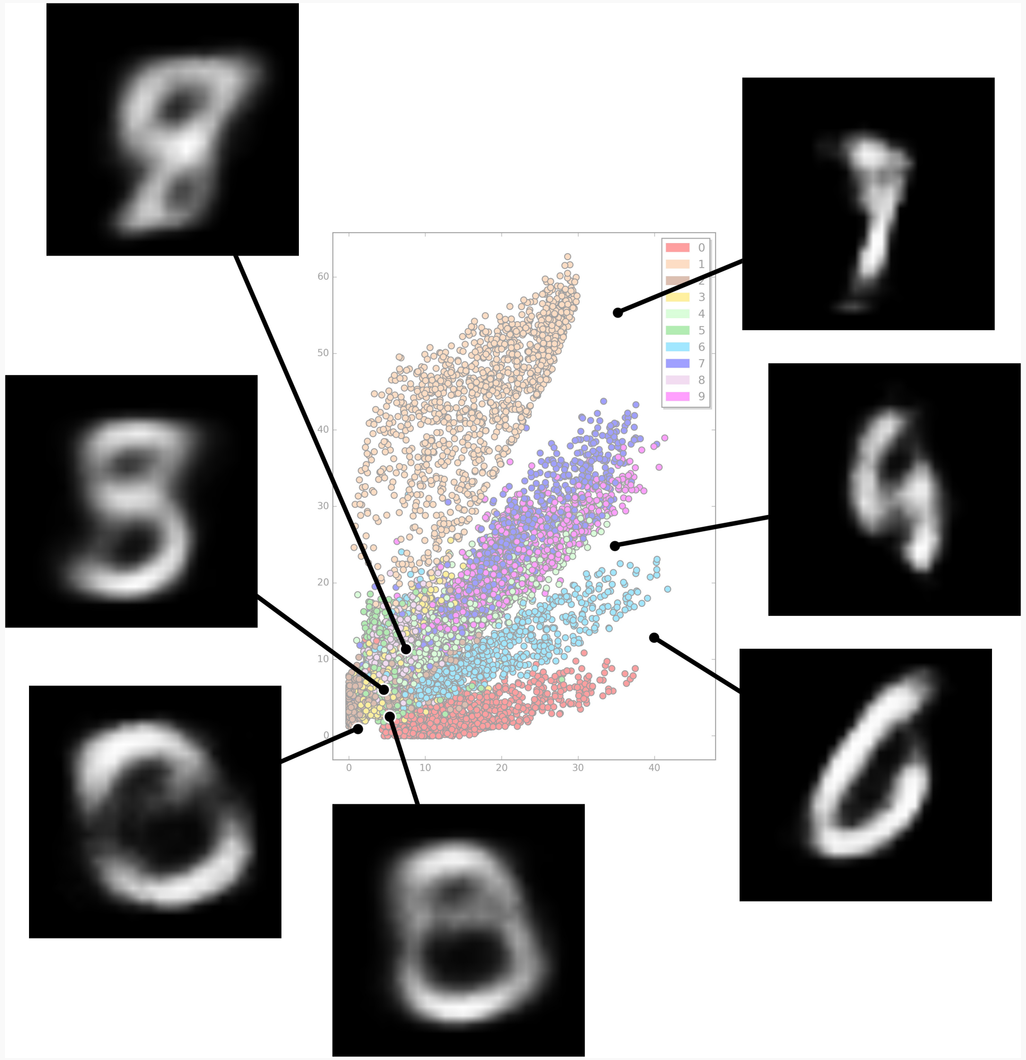


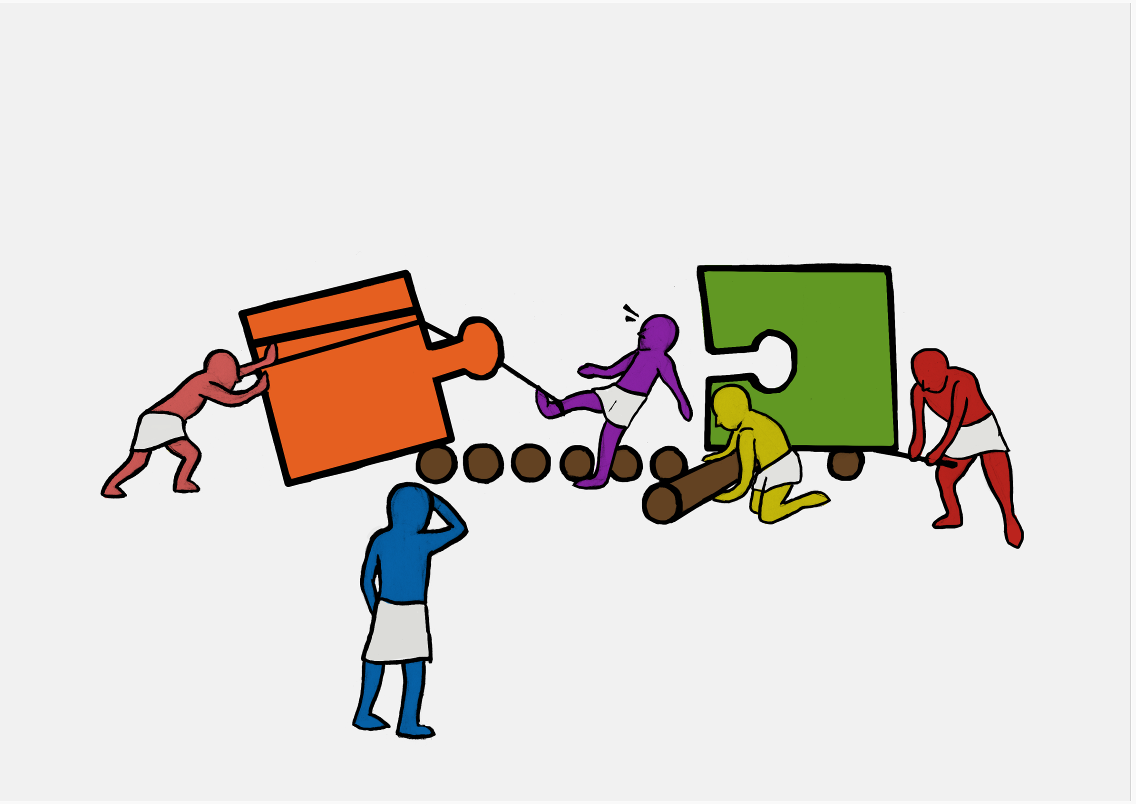
Image taken from A. Glassner, Deep Learning, Vol. 2: From Basics to Practice

Problems with Autoencoders

- Discrete latent space
- The latent space contains a lot of gaps (separability)

The cure:
Variational autoencoders (VAE)





Exercise 2:
Recreating an image of Pavlos