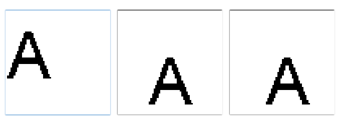
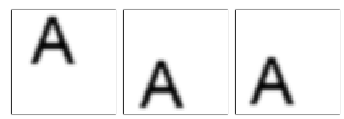
TOY EXPERIMENTS

This document provides several toy experiments to compare the different features learned by CNN-based models and implicit-decoder-based models.

A. Training data

The training dataset is synthesized by putting binarized letter A's on white background. The image of A is 29x27 (height x width), and the image size for the dataset is 64x64, therefore we have 35x37 possible positions to put the whole A in the image, and that makes our dataset contain 35*37 = 1295 images.

We have two versions of the dataset, **A** and **A_blur**. **A_blur** contains blurred A's which are Gaussian blurred with sigma = 1. See below some samples from each dataset.

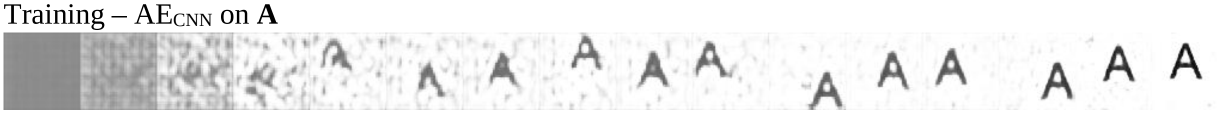
Dataset **A**:  Dataset **A_blur**: 

Besides, we have incomplete versions of the above datasets. We removed the middle four rows of possible positions of A, resulting in 31x37 possible positions. We call the incomplete versions **A_gap** and **A_gap_blur** respectively. Such dataset can test the model's ability to generate unseen shapes to fill the gap.

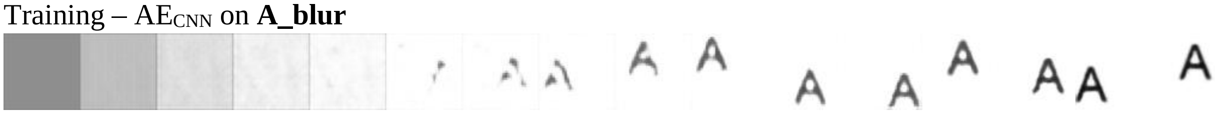
B. Autoencoder (AE)

We trained AEs with CNN decoder and implicit decoder on **A** and **A_blur**. Below we show AE reconstruction samples during the training process. We will use subscripts X_{CNN} and X_{IM} to distinguish different decoders throughout this document.

Training – AE_{CNN} on **A**



Training – AE_{CNN} on **A_blur**

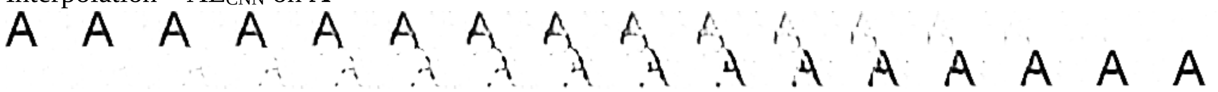


Training – AE_{IM} on **A**

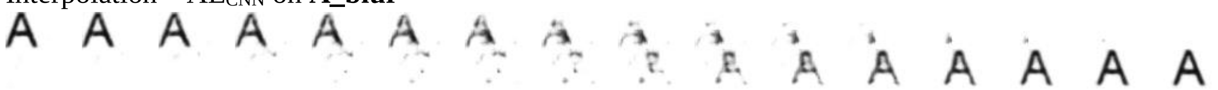
Training – AE_{IM} on **A_blur**

One can get a feel about the features learned by different decoders. AE_{CNN} is predicting the possibility for each specific pixel to be on or off, while AE_{IM} is learning the field of the shape. Below we show AE interpolation results, to show the movement is learned by AE_{IM} while AE_{CNN} is simply memorizing each specific shape.

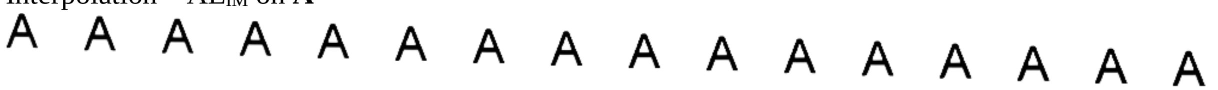
Interpolation – AE_{CNN} on **A**



Interpolation – AE_{CNN} on **A_blur**



Interpolation – AE_{IM} on **A**



Interpolation – AE_{IM} on **A_blur**

C. Variational Autoencoder (VAE)

We trained VAEs on **A_gap** and **A_gap_blur**. VAE constraints the latent space to be compact, therefore the interpolation with VAE is much better for CNN based models. But can those models generate unseen new shapes to fill the gap? Below we show VAE interpolation results. We use three rows for each interpolation sequence: the first row is before the gap, the second is crossing the gap, the third is after the gap. Notice the fractured shapes when VAE_{CNN} is crossing the gap. We found that it is better to show the result in an animated way, therefore we provide animated GIF images in the folder. One can open GIF with their image viewer or web browser.

Interpolation – VAE_{CNN} on **A_gap**

Interpolation – VAE_{CNN} on **A_gap_blur**

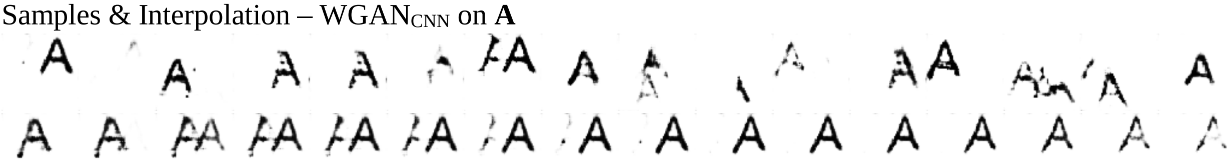
Interpolation – VAE_{IM} on **A_gap**

Interpolation – VAE_{IM} on **A_gap_blur**

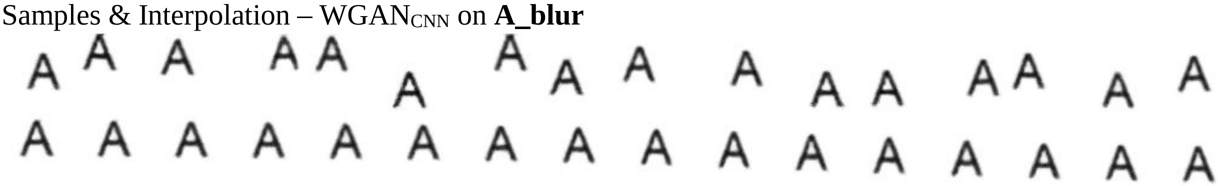
D. Wasserstein GAN (WGAN)

We did not train WGANs on **A_gap** and **A_gap_blur**, since the discriminator may consider shapes crossing the gap as negative samples and prohibit the generator from generating such samples. We trained WGANs on **A** and **A_blur**, to show shape movement is not continuous in the output space of CNN decoders. We use two rows for each set of samples: the first row is generated samples with random initialized latent vectors, the second row shows an interpolation sequence.

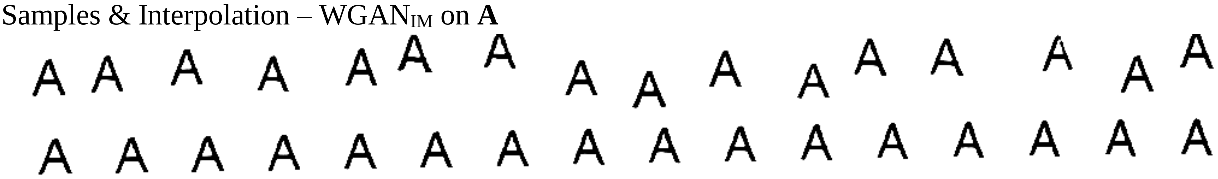
Samples & Interpolation – WGAN_{CNN} on **A**



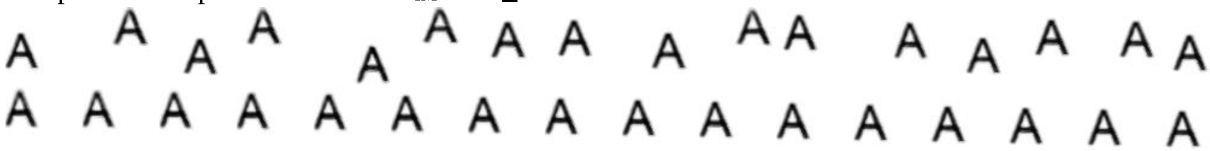
Samples & Interpolation – WGAN_{CNN} on **A_blur**



Samples & Interpolation – WGAN_{IM} on **A**



Samples & Interpolation – WGAN_{IM} on **A_blur**



The perception of continuity for CNN decoders is the lightness change in each pixel, therefore dataset **A** is far from continuous in WGAN_{CNN}' eyes. A direct consequence is that WGAN_{CNN} cannot be trained well on dataset **A**. But if we blur the images to provide such lightness changes and make the dataset continuous in WGAN_{CNN}'s perception, it will get good results, see the results from WGAN_{CNN} trained on **A_blur**.

As for the implicit decoder, both lightness changes and shape movements are continuous, therefore it can be trained well on both **A** and **A_blur**.