Assignment 1

Batch Size of 3

December 17, 2023

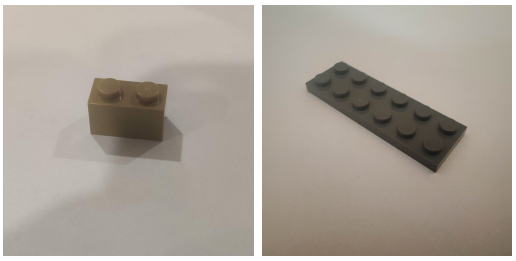
1 Data Collection

1. Data Collection

   - The data is collected in various ways, we have two different datasets, one of which is taken by a camera of various lego pieces, the other is computer generated and rendered. We chose to collect lego pieces due to the fact that it is unique, creative, and scene relative. Not only does this allow the model to generating an image based on the contents around it, but also include the image relative to the scene. Moreover, having different geometric shapes, helps it understand the locality of the scene. This dataset is also an easy to define dataset, in which we can label and classify without too much risk for false classifications or biased classifications (i.e. red lego is red as opposed to this person is a "drunkard").

2. Data Curating

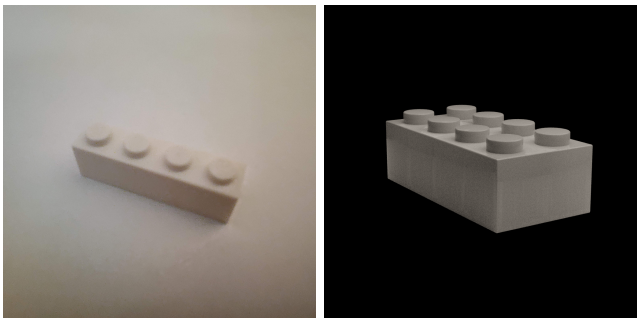
   - Photographed Legos:



(a) Old Image (b) New Image

     – Here are two examples of a previously generated image (a) and a newer image (b). We had previously generated a past dataset that has around 500 images, this dataset included (a). However, there are problems to the images: poor lighting, occluding shadows, and a pattern of a certain shadow (phone).

   - Computer Generated Legos:



(a) Photographed (b) Computer Generated

     – Here are two examples of a photographed image (a) and a computer generated image (b). As you can see, the computer generated image is cleaner, has no background effects (shadow), and clearly represents the lego in the correct space at the correct angle at all times. We used both datasets in our VAE.

3. Data Augmentation:

- Data augmentation pipeline: (image) ⇒ (mirror) ⇒ (greyscale)
- In our original two datasets, we did not augment them in the physical file. When the images are placed into a dataset on python, we augmented the images by mirroring them. This allowed us to double our dataset size and still maintain a bit of a uniqueness.
- We did not choose to rotate or flip due to the fact that a flipped lego piece or rotated lego piece will confuse the model based on dimensional and perspective. That means, when the VAE attempts to learn the shape and perspective of the general lego piece, it will have a harder time trying to understand in which axis it should be in the right position. For example, if we have a lego rotated 30 degrees, should it be compared to that way or should it be compared to as if it were on the ground?

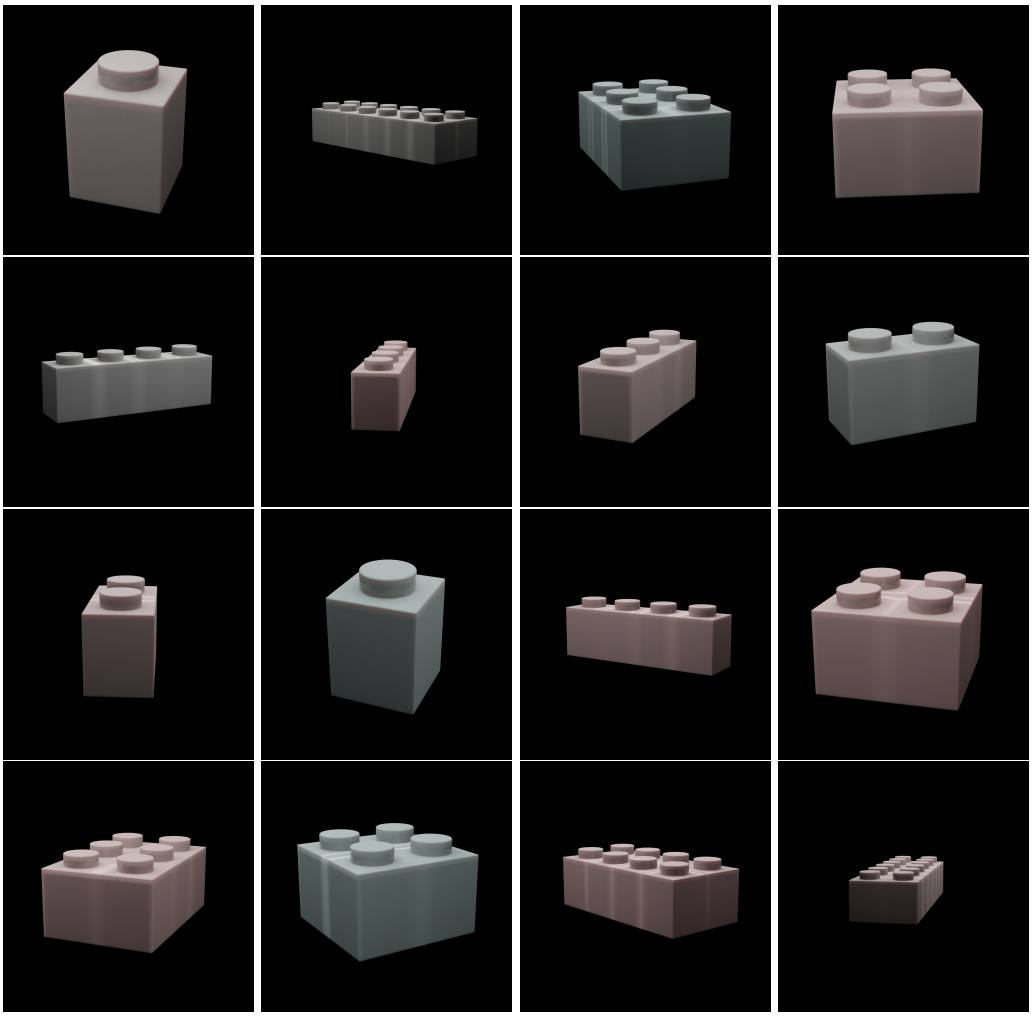


Figure 3: Example of the computer generated dataset

Figure 4: Example of the photographed dataset

2 VAE

a.

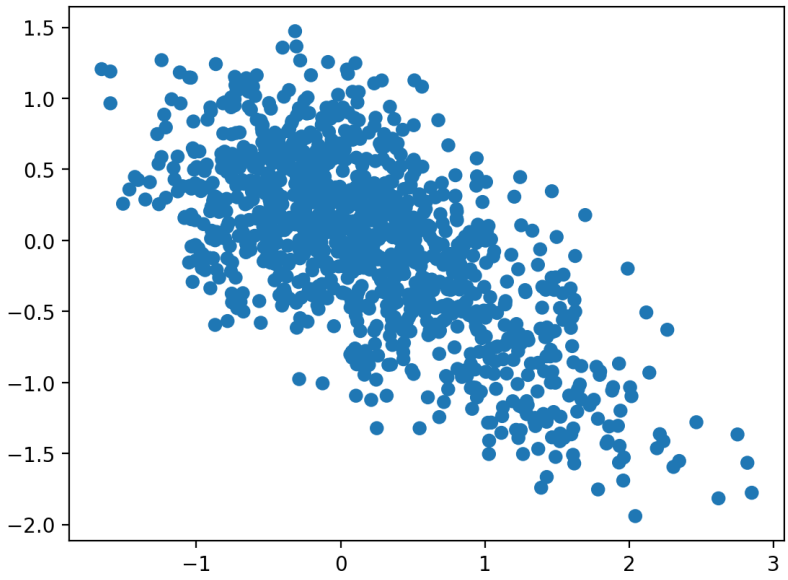
1. The pipeline fore this project is: 1024x1024 Images → Resize to 64x64 → Augment the data by adding mirrored images to the dataset → Train VAE → Plot the images across the latent space.
2. Vanilla VAE
3. 2D latent space was chosen since it is easy to show the latent space and do latent space arithmetic or transformations.
4. For the real life legos, converting and augmenting data takes about 1 minute and training takes 14 seconds. For the rendered legos, 1.5 minutes and 1.7 minutes.
5. I was unable to do 1024x1024 training since I didn't have enough vram for it, but for the repo that I cloned, the highest dimesion that was able to still be graphed was 64x64. So we had to lower the resolution and we decided on 64x64 since it was still relatively fast to train and it was the biggest we can do.
6. The biggest challenge was trying to get the dataset to actually work with the VAE that was found. What was really helpful was looking at the shapes of the numpy arrays if you get operation errors. In addition, I wasn't sure how to get the graphs working with colors so I decided to do only greyscale instead.

b. Passing Original Images Through VAE (Real Legos)

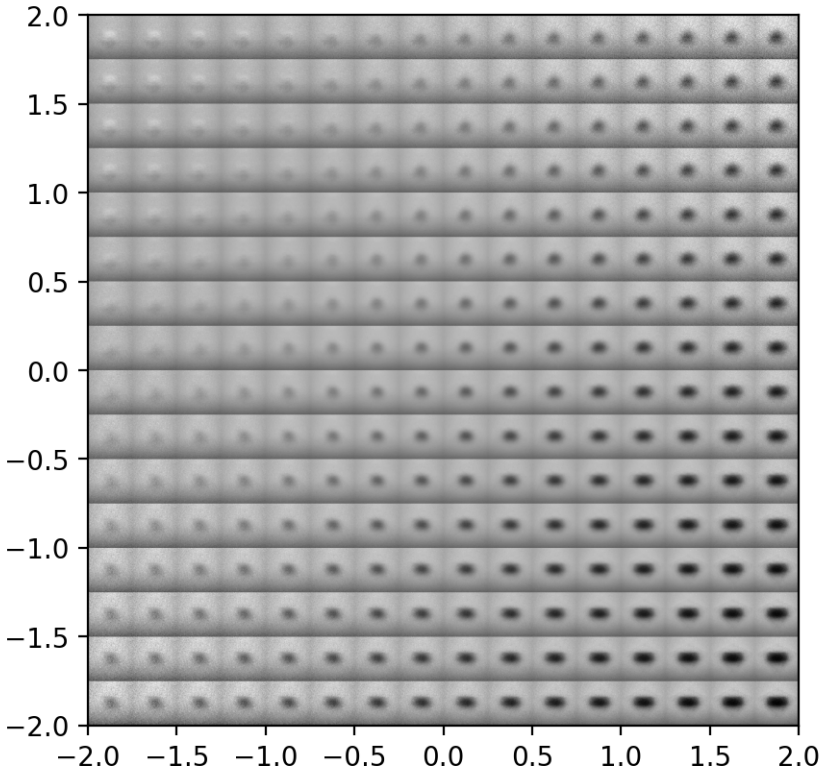
The VAE managed to capture the color of the pieces, however the shape of the pieces were lost. There is one outlier which is the large float white piece; that piece's color and shape were not captured.

c. Graphs of the Latent Space (Real Legos)

Here is a plot of the dataset on the latent space:



Here is a plot of images sampled across the latent space:

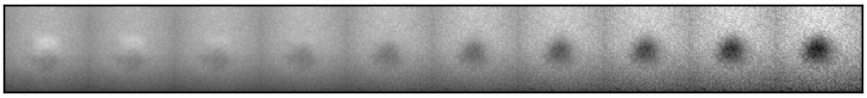


The area nearby the origin in latent space is very blurry and hard to distinguish between the lego and the background. The farther out you go from the origin, the more the lego looks like but there is more noise added to the image.

The two features that the VAE were able to pick up on were the color of the lego and a very general shape or rotation of the piece.

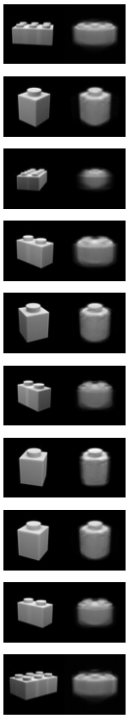
I think the images are very reasonable given the data.

d. Interpolation of Two Places in Latent Space (Real Legos)



I took two points in the latent space (-2, 2) to (2, 2) and interpolated the space between these points with the VAE.

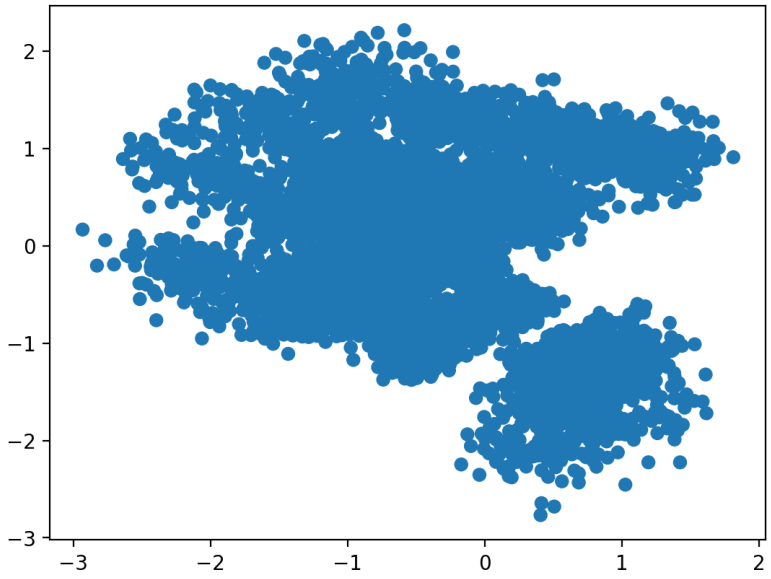
b. Passing Original Images Through VAE (Rendered Legos)



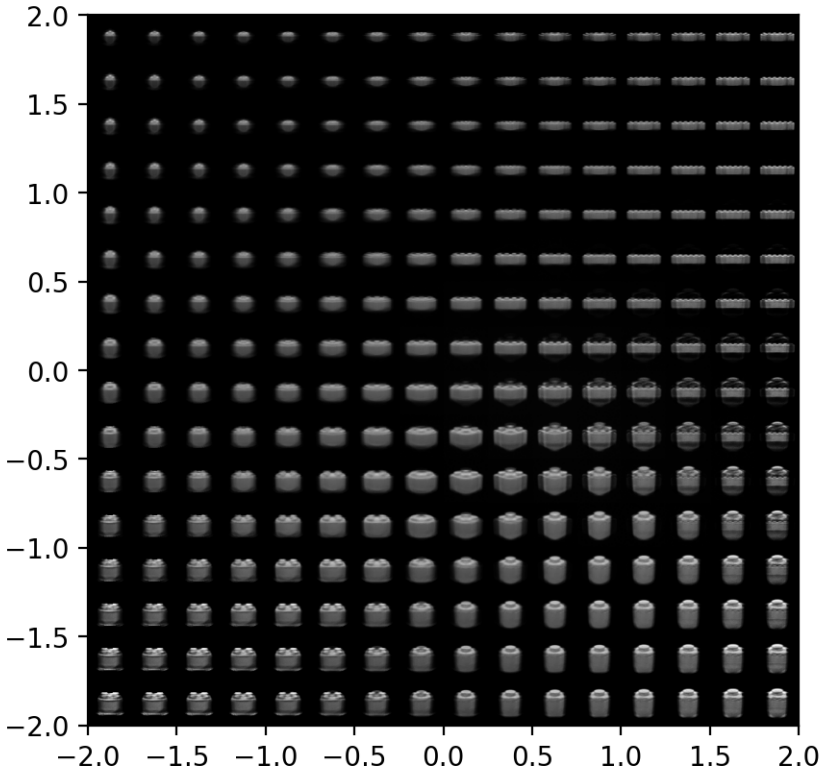
Compared to the real legos, the VAE managed to better capture the overall shape of the legos. This is probably due to the more consistent framing of the legos in the center of the image. However, some of the legos when look blurry as if it VAE were rotating the lego.

c. Graphs of the Latent Space (Rendered Legos)

Here is a plot of the dataset on the latent space:



Here is a plot of images sampled across the latent space:

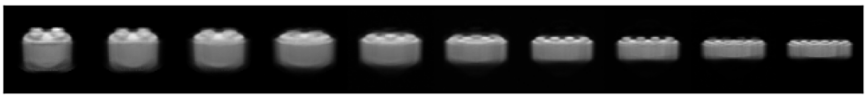


Compared to the latent space of the real legos, there are clear areas where different types of bricks are located in the latent space. Like the other VAEs that were demonstrated in the classroom, thee are regions in the space where the VAE is blending between two or more different types of bricks.

The features that the VAE were able to pick up on were the type of bricks (1x1, 2x2, 2x4, 2x4). Since there wasn't very much in difference in the colors of the bricks, they are all the same greyish color.

I think the images are very reasonable given the data.

d. Interpolation of Two Places in Latent Space (Rendered Legos)



I took two points in the latent space (-1, -1) to (1, 1) and interpolated the space between these points with the VAE.

3 Project Proposal

(a) What are the effects that you hope to create in your project? How do you think the effects will support the narrative of the video? (that is, why are the effects you propose suitable to the context?)

- Our problems are object style transfer with captioning. Style transfer takes a content and a style, then transform the content due to that style. A "style" in this context refers to both the geometry of the input object and its textural data. Captioning takes a scene and generates a short text description based on all the content within the scene. Our project idea is to use a scene, which contains multiple 3D objects, and perform style transfer, texturing and captioning.

(b) A storyboard that walks us through what we will see as the video progresses.

- Begin with an overview of the original scene, containing various objects inside of a room, such as desks, chairs, a bed, bookshelves, books, plushie, snacks, etc.
- Focus to specific parts and for each object within the scene:
  (a) Perform geometric style transfer.
  (b) Apply textural style transfer.
- Repeat for multiple style inputs. The first cycle will showcase the fact that the program will apply the style transfer individually to each object, while latter cycles can simply show a before and after transformation process in quick succession.
- Caption the scene continuously as objects get transformed.

(c) How are the effects related to one or more of the concepts in the book/course? What other related work is there from the technical or artistic literature that is related to your idea? (research papers, examples of similar effects in videos you've seen).

- The main concept for this is to experiment with the creative elements of augmentation. As we've seen in class, there are various ways for artists take an original concept, image, or model, and transform it to its own style.

  For instance, in class we referenced Hertzmann's Painterly Rendering, where images are taken and algorithms are used to transform that original image into a painting of that image. Therefore, we plan to do the same with 3D models, taking an original model and transform the style of that model to match whichever style we want it to look like. "Style" would constitute as the general shape of an object and micro-patterns found on the surface of the model.
- For style transferring on the 3D models, we will primarily use the 3DSNet research paper. For transferring of textures, we will use StyleRF. For generating captions, we will use ObjectNet3D as part of the pipeline to recognize objects within the scene.

(d) Where will you collect your raw image/video/audio data from?

- We will generate our own models by creating scenes in blender. We will then convert the mesh into a point cloud format so that we can have it as an input to the ML models.

(e) What software do you anticipate using to accomplish your goals? What type of coding will need to be done to accomplish the effect?

- We will be studying 3DSNet for style transfer and ObjectNet3D for object classification and using provided code in combination with custom code to produce the models. We will train these models on the ModelNet dataset and our custom datasets.

(f) What is your strategy/plan of action for realizing your conceptual idea? That is, what tasks need to be performed? What is the rough schedule for these tasks?

- Reproduce the 3DSNet paper.
- Fine tune on our custom dataset and perform testing.
- Write model for converting scene into point cloud, performing clustering and object identification.
- Perform style transfer on each object within the scene.
- Use StyleRF to add texturing to objects.
- Generate captions on completed scenes with ObjectNet3D and other relevant tools.