CS5955 Final Project: Text to Image Synthesis

Ian Argyle & Daniel Bartz

December 10, 2021

1 Introduction

Text to image synthesis is an interesting domain of deep learning which combines two areas which are often treated separately in the field, with differing techniques employed: text analysis and computer vision.

Earlier techniques such as Image GPT [2] seem to focus more on learning pixel values directly. However, techniques like this require very large models to generate high resolution images, and often this isn't feasible.

Newer techniques instead focus on learning smaller distributions which can be converted into an image. For example, the paper "Taming Transformers for High-Resolution Image Synthesis" [1] proposes using a VQ VAE to learn a small (1x16x16) latent space representation of images, then training a transformer on that latent space to sample new images. This technique also incorporates the use of a CNN discriminator and a perceptual loss to generate higher quality image reconstructions.

We originally intended to train a VQ VAE and some kind of text autoencoder to learn a shared representation of the latent space between the images and corresponding labels by using MSE loss between the latent spaces, plus a reconstruction loss for each autoencoder. Which we could then use to generate new images by encoding the text, then decoding the corresponding image. However, after reading [1], we decided to follow their paper more closely.

2 Approach

We decided to deviate from [1] in two major ways. First, rather than using a transformer to learn the latent space for the sole purpose of sampling new images, we trained a transformer to translate text into the image latent space. So, in order to sample a new image, a prompt must be provided. Second, rather than using both perceptual loss and a discriminator, we chose to use only perceptual loss.

Our basic model architecture consists of a VQ VAE and a transformer. The VQ VAE accepts images of size 3x128x128. The input image is first sent through a singe 7x7 conv layer, then fed into a series six of residual encoder blocks which each consist of 4 3x3 conv layers with batch normalization and ReLU in between. Some encoder blocks down sample, down sampling happens three times, meaning the output of the encoder is nhidx16x16 where nhid is the number of conv filters (256 in our case). Then, the exact opposite happens in the decoder, where the image is up sampled three times using conv transpose layers.

The intuition behind our choice of using only perceptual loss is the following: both discriminators (in GANs) and perceptual loss ultimately serve the same purpose of helping the generator (or decoder in this case) learn to generate higher quality images by using a separate model. We assumed that choosing the right layer (or combination of layers) in our perceptual loss network would result in high quality image outputs. Specifically, we wanted to choose a layer that (1) provides the highest MSE loss between dis-similar images, and (2) provides the lowest MSE loss between similar images. In this case, similar images can be a relatively loose term in the sense that we want the image output of our VQ VAE to similar in content to the input, but not necessarily similar pixel wise. After all, we are trying to model a latent space that contains contentual information, not necessarily the locations of specific pixels. To this end, we chose ResNet-18 [4] as our perceptual loss network, and graphed the MSE loss between a few sample image pairs at each layer. The results are below:

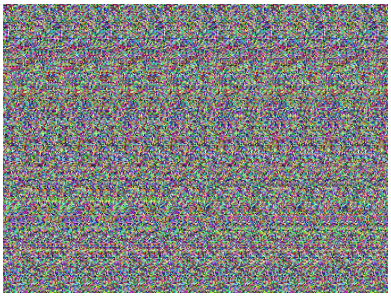
(a) Beach 1

(b) Beach 2

(c) City

(d) Airplane

(e) Dog



(f) Noise

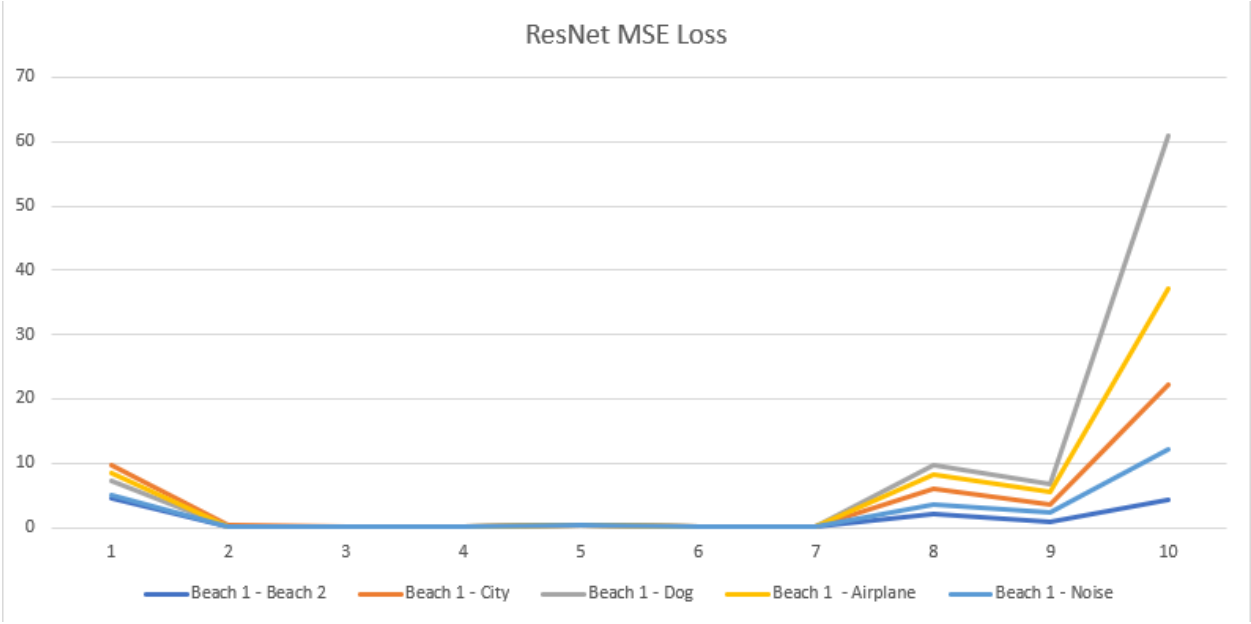


Figure 1: MSE loss between Beach 1 image and other images for each layer in ResNet 18

As we would expect, the loss between the beach 1 and beach 2 images is consistently the lowest, and the most deviation between losses appears to be after the last layer, which is the classification linear layer. However, the loss between beach 1 and the noise image is the second lowest, which is concerning. Given that the last layer has the most deviation, and that we're trying to learn content, not pixels, we decided to try training a VQ VAE with reconstruction loss defined as the MSE loss at the last layer of ResNet. Unsurprisingly, the results were not good. The VQ VAE converges, however the images it generates look like noise. Presumably the VQ VAE is learning to "trick" the ResNet, resulting in images that get classified a specific way, however look nothing like actual images. Something akin to adversarial attacks.

Armed with this new information, we decided to replace the last layer loss with that of an earlier layer, specifically the second layer of ResNet 18. The results of this were better, the VQ VAE seems to be able to reconstruct general shapes, however the results were still extremely noisy. After testing, we decided to use MSE loss directly between the original image and the reconstructed one. This provided much better results:

(a) Original Image (b) MSE Reconstructed (c) Perceptual Reconstructed

Figure 2: VQ VAE Reconscruted Images

Next, we began working on the text transformer. The basic architecture is as follows: tokenized sentences are first converted to learned word embeddings, fed through the transformer, which then produces a distribution of likelihood over the size of the vector quantizer vocabulary (in this case, 512) in order to generate a hidden latent space tensor corresponding to the "language" of the VQ VAE. This tensor can then decoded into an image using the VQ VAE decoder. Loss is calculated using MSE loss between the latent space tensors for corresponding text image pairs.

Due to time constraints we were not able to test the transformer (the VM crashed as it was saving the model, no time to retrain). So we do not know how well this approach would've worked, however the transformer seemed to be converging, so it's possible.

3 Future Work

We intend on integrating this project with our capstone project for the purpose of generating images for social media posts. Specifically, we are using GPT-3 [3] to generate text content for social media posts based on context such as business name and description. We think we can use GPT-3 to generate image labels, then use this model (or something similar) to generate a corresponding image. However, we have a long way to go before that point.

In the future, we'd like to continue fine tuning this approach to generate reasonable images. Specifically, we would like to experiment with using a combination of different layers of ResNet (or another model) for our perceptual loss. Also, we believe GANs are the way to go for image synthesis, so we would like to implement a discriminator similar to the one described in [1]. For most applications, 128x128 images are not high enough quality, there have been a few approaches designed to generate higher resolution images without too many added parameters (like in [1]), one straight foreword approach would be add more up sampling layers in the decoder network, then perform MSE loss between that output and the original image before it

was down sampled, or at least a higher quality version of it. This however adds parameters and complexity, so we will likely explore other approaches.

Overall, we may not have completed the task we originally set out to do, but we learned a lot from trying things, and will continue to do so in the future.

References

[1] Patrick Esser, Robin Rombach, Björn Ommer (2021) *Taming Transformers for High-Resolution Image Synthesis*, arXiv:2012.09841

[2] Hanting Chen, et al (2020) *Pre-Trained Image Processing Transformer*, arXiv:2012.00364

[3] Tom B. Brown, et al (2020) *Language Models are Few-Shot Learners*, arXiv:2005.14165

[4] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun (2015) *Deep Residual Learning for Image Recognition*, arXiv:1512.03385