Image Generation with Prompt-based Control via VQ-GAN and Transformers

Michael Zeolla [1] **Jacob Reiss** [1] **Mario Zyla** [1] **Ningcong Chen** [1]

Abstract

Image generation is a novel concept in the areas of Deep Learning, which has resulted in many different unique model architectures with the goal of simple and advanced generation. However, many such models suffer as the dimensions of the image space expand due to the increased computation costs; and conditional image generation boasts an entirely new set of problems.

This project builds upon the established VQ-GAN and Transformers architectures to handle advanced Conditional Image Generation where prompt-based directions can control the output. This is accomplished via an LSTM + Transformer combination, along with a VQ-GAN.

1. Introduction

Computer Vision is one of the many domains of deep learning, and while numerous advancements have been made in the realm of understanding images, and generating them; only recently have research efforts begun to bear fruit for high-resolution conditional image generation.

A common issue with VAEs trained on latent spaces is that they are prone to posterior collapse (Oord et al., 2017); This happens when the model fails to learn useful information about the input data; which results in both poor image quality and a lack of diversity in the types of images generated.

A novel approach to solving the issue of posterior collapse is the concept of a Vector Quantized-Variational Auto-Encoder (VQ-VAE) (Oord et al., 2017). Using vector quantization, a discrete latent space can be represented as a finite set of vectors. This approach doesn't sample from the pixel space, but rather the learned latent.

Additionally, Transformers are known for being powerful at learning long-range sequences in data. However, as the length of the input increases, so do the computing costs, making high-quality image generation incredibly difficult. One way around this is by using CNNs and their inductive bias to prioritize local sequences over long-range sequences. An implementation of this is the VQ-GAN with a transformer model (Esser et al., 2020). The VQ-GAN encodes important information from the input data which the transformer can then generate images from.

1.1. Research contributions

This project aims to bridge the gap between conditional image generation reliant on more technical input, and those more applicable for everyday usage. For example, conditional image generation via segmentation masks or class labels are two common examples (Esser et al., 2020). However, people most often use language to describe images, such as the phrase "A yellow Daisy", and as such, previous implementations were severely limited in their everyday usage. A more appropriate approach for conditional generation would incorporate language. This paper explores that idea by implementing an architecture for understanding language and using prompts to generate samples 3.

2. Related Work

The field of image generation has advanced significantly with the introduction of techniques such as generative adversarial networks (GANs) and variational autoencoders (VAEs). This section outlines some key methodologies that the team incorporated in the study, focusing on the developments in VQ-VAE/VQ-GAN and Transformers, and their applications in image synthesis.

2.1. VQ-GANs

Generative Adversarial Networks (GANs) are another approach to image generation. Radford et al. (Radford et al., 2016) showed that deep convolutional GANs (DCGANs) could create high-quality images by improving the structure of both the generator and discriminator. However, GANs can be difficult to train and fail to represent diverse inputs, a problem known as mode collapse.

To enhance the quality of image reconstructions further, Esser et al. (Esser et al., 2020) proposed VQ-GAN, which

[1]Worcester Polytechnic Institute. Correspondence to: Michael Zeolla <mjzeolla@wpi.edu>, Jacob Reiss <jtreiss@wpi.edu>, Mario Zyla <mazyla@wpi.edu>, Ningcong Chen <nchen3@wpi.edu>.

CS/DS 541 Deep Learning Final Project, Worcester Polytechnic Institute.

merges the stable training dynamics of GANs with the effective latent space quantization of VQ-VAEs. VQ-VAE introduces a latent representation that is better at capturing the variations in data. It also addresses a common issue in standard VAEs known as "posterior collapse," where the model overlooks the latent code. VQ-GAN builds on VQ-VAE by adding a GAN-based discriminator that helps the decoder produce more detailed images.

2.2. Transformers for Image Synthesis

Originally designed for processing text in natural language tasks, Transformers have recently been adapted for image processing. Parmar et al. (Parmar et al., 2018) introduced the Image Transformer, a model that applies the self-attention mechanism to image generation. This model shows how Transformers can handle long-range dependencies in image data. Additionally, Chen et al. (Chen et al., 2020) demonstrated that Transformers could generate coherent images by training on sequences of pixels, showcasing their flexibility in dealing with images.

Despite the advancements with VQ-GANs and Transformers in image synthesis, their ability to integrate complex, natural language descriptions into the generation process remains limited. The project explores this gap by enhancing the conditional capabilities of these models, allowing them to generate images from detailed text prompts, not just simple labels or metadata.

3. Proposed Method

3.1. Learning the Latent

The first step in the process of conditional image generation is to properly learn an embedding space for representing images. Generating images from the pixel space is computationally expensive. This is because as the number of pixels increases, so does the cost of sampling in the pixel space (Chen et al., 2016). Therefore, an image embedding helps reduce the computational cost of sampling, while retaining the original image structure. The chosen encoding model to accomplish this was the VQ-GAN architecture, which consists of a CNN encoder, CNN decoder, and a GAN discriminator (Esser et al., 2020).

As seen in Figure 1, the main idea behind the VQ-GAN architecture was to learn the latent representations of the pixel space via codebook vectors. These vectors can then be decoded to generate a reconstructed image from the learned latent; which matches the original image.

The CNN encoder model included a series of downsampling blocks that aim to reduce the dimensions of the input images, stripping them down to the most important features, along with normalization layers. The CNN decoder

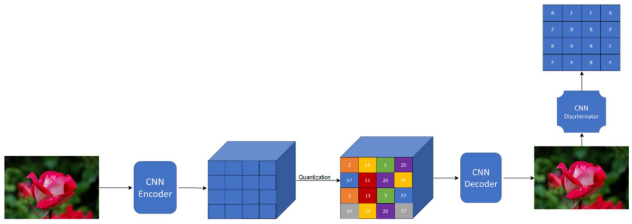

Figure 1. VQ-GAN Architecture

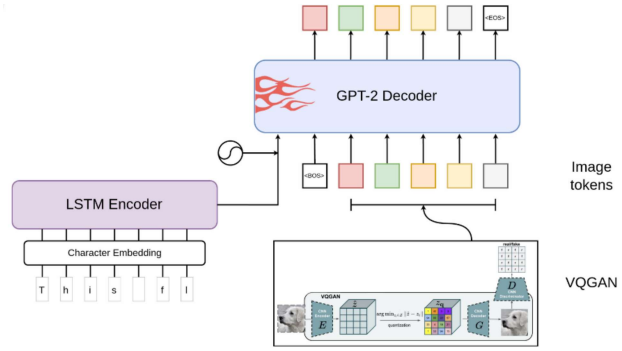

Figure 2. Conditional AutoRegressive Transfomer

includes multiple upsampling blocks in the form of Convolutional Transposition layers, which reconstruct the image from the codebook vectors. Finally, the discriminator follows a PatchGAN implementation (Isola et al., 2017). The amount of downsampling applied was a configurable field and directly correlated with the dimensions of the dataset.

3.2. Applied Transformers for Image Generation

After training the VQ-GAN architecture to act as an encoder, images can be easily mapped from pixel space to codebook space. To do so, the model must simply intake an image, pass the image through the encoder, and finally map it to the appropriate codebook vectors via quantization. Then the codebook vectors from the encoded image are transformed into a flattened array of codebook indices, where an index represents the position of the codebook vector in the latent space.

Basic Image Generation

The codebook indices can be passed into a Decoder-Only Transformer for token prediction. The task at hand is to predict the next codebook index. At each step, the transformer outputs a probability distribution for the next possible codebook index - allowing for a new vector embedding to be added to the sequence. The generated codebook tokeens are then unflattened and mapped back to the original codebook values via the index. The codebook vectors represent an encoded image from the VQ-GAN architecture; which can then be decoded into a new image.

Prompt Based Conditional Generation

To enforce prompt-based conditional generation the transformer model needs a method for understanding each image in terms of human language, along with the original image input. A new embedding model, Long-Short Term Memory (LSTM) Model is required to capture the embed context for each caption. The phrase "A long red car" can be treated as a series of character tokens, and be encoded into a numerical vector. The original codebook sequence can then be concatenated with its respective contextual text embedding and passed to the transformer together as input. From then onward, the transformer would begin to learn the association for each codebook vector regarding the input text. To generate a new conditional image, the transformer must be provided with a similar text embedding, from which it can begin to produce a related codebook.

4. Experiments

During the experimentation phase of the research project, the team underwent a wide range of changes to improve the model. These changes included variations to the different model architectures but also included testing with different hyperparameter sets. In the end, all the different experiments conducted helped to boost model performance and the generative capabilities of the architecture.

4.1. Dataset Selection

For the project, selecting the right datasets was crucial to address different aspects of image generation. The team chose to work with a variety of datasets, each offering unique challenges and learning opportunities for the models. While there were many datasets applied during the training and testing phases of the project, the **Oxford 102 Flower Dataset** was the main implementation baseline. A full list of all the dataset configurations is available at A.1.

4.2. Experimenting with the Latent

The team experimented with a wide range of model architectures to output the best-reconstructed images. The loss function for the model was a combination of the original VQ-VAE loss, with reconstruction loss, commitment loss, and codebook loss; along with the additional GAN loss for the discriminator (More Info A.2). The loss function learns the correct codebook space by forcing image embeddings to cluster toward the codebook vectors. Additionally, the GAN loss includes a λ hyperparameter which helps balance the PatchGAN and VQ-VAE loss (Esser et al., 2020).

In regards to model architecture, the team experimented with the overall size and complexity of the model. The main experiments for the VQ-VAE model focused on the number of layers present, along with the size of each layer. For example, the team experimented with the number of Downsampling Layers for the encoder/decoder, which directly impacted the resolution of the reconstructed image. Additionally, the model supported adding additional encoder/decoder heads, increasing model capacity.

4.3. Generation with Transformers

The Autoregressive Transformer followed the typical cross-entropy loss function, where the loss was computed based on the probability distribution when generating the next token (More Info A.2). Additionally, the loss function included a conditional component based on the input.

For the transformer architecture, the team experimented with many of the different hyperparameters associated with training. One hyperparameter was the total number of transformer attention heads, which directly correlates to how many parameters are trained in the model. Furthermore, the number of layers within the transformer was another point for experimentation. Lastly, the team also tested the number of input tokens to the transformer and the total embedding size of the images.

4.4. The Sequence Length Problem

In the aforementioned transformer model architecture, codebook indices are provided to the transformers as tokens, acting as the predictor; and this is done by passing the entire encoded codebook sequence for the image as input to the transformer. However, this approach does not scale for higher-resolution images since the transformer attention mechanism places limits on the input sequence size. To help mitigate this issue, the team experimented with the sliding window technique (Esser et al., 2020).

Similarly to Convolution Operations in CNNs, the sliding window technique operates by sliding a frame over the entire input image. Instead of passing the entire input to the transformer, only those codebook tokens accessible in the frame are passed to the transformer. This technique helps limit the number of input tokens for the transformer.

4.5. VQ-VAE Attention

Attention is a mechanism that allows for context to be shared across an input space; learning long-term dependencies. Since high-resolution inputs depend on large input spaces, the model must be capable of learning such long-term dependencies. One additional modification to improve results was to add VQ-VAE attention to the architecture (Esser et al., 2020). When attention is considered the model is augmented to include additional Key, Query, and Value matrices, which are applied to the input. For the VQ-VAE model, the weights are stored in a CNN layer and applied via the attention formula (Vaswani et al., 2017).

Figure 3. VQ-GAN Reconstruction Results. Top: Original, Bottom: Reconstruction

Figure 4. Class Conditional Generation on Oxford 102 Flower. Left: Generated, Right: Original

5. Results

5.1. VQ-VAE Encoding/Reconstruction Results

The team first implemented the original VQ-VAE architecture, which did not include the GAN augmentation (Oord et al., 2017). This model provided a good starting point for experimenting with image reconstruction, without unnecessary complexity. However, many of the reconstruction results from the simple VQ-VAE model were blurry approximations. The team experimented with different sets of codebook encoding sizes, and residual layers, but did not see any major improvements. The results from VQ-VAE can be seen in Figure 9, which shows a blurry reconstruction of the flower dataset.

Through experimentation, the team found that the downsampling layers in the original VQ-VAE model are too simple: one downsampling layer has only a convolutional layer, followed by an activation function and a batch normalization layer. Many of the image details were lost during the downsampling step. Therefore, the team adopted an alternative design that could help to prevent such a problem; VQ-VAE: UNet. Compared to the original VQ-VAE model, VQ-VAE UNet inserts Residual blocks between downsampling, with upsampling layers in the decoder (Ronneberger et al., 2015).

With the new VQ-GAN architecture, based on UNet, the model was more optimized to learn a rich codebook latent space. The Taming Transformers for High-Resolution Image Synthesis paper also applies a dynamic coefficient for λ. In practice, the team noticed that the loss curve of GAN loss was not smooth, and such a strategy may lead to unstable training. Therefore, the team applied a static coefficient and empirically set its value to be 5e-4. The reconstruction result on the FFHQ face dataset and flower dataset (3) provided significantly better results than the original VQ-VAE model. For example, the images are far less blurry, and capture a higher degree of detail in each image. However, even with the additional GAN architecture, the model still experiences loss in the finer details of input images. Figure 11 shows an example where the details are missed by the reconstruction. Specifically, the original wrinkles are smoothed out (caused by VAE) and the adversarial effects (caused by GAN) on the finger. The results on alternative datasets can be viewed in Figure 10.

For all the experiments shown in the paper, the team applied a codebook of size 256 and a downsampling factor of 4, and the input images were resized to a shape of 64x64. Therefore, the latent codebook space is 32x32.

5.2. Conditional Image Generation Results

Pixel Space Class-Conditional Generation

The team first attempted to perform small-scale image generation in the pixel space, as opposed to the latent space learned with VQ-GAN. For most image generation tasks, sampling from the pixel space is ineffective and costly, since it relies on the dimensions of the input image. The main reason for pixel space generation being implausible for everyday usage is because training time scaled linearly when working with larger dimensional spaces. For example, if it takes one hour to train on MNIST (28x28x1) then it will take 15x longer to train with the ImageNet (64x64x3) dataset. With that in mind, it was only feasible to perform pixel space generation on smaller datasets, MNIST and Fashion-MNIST. The transformer for the pixel space generation results in Figures 13 and 14 is an embedding size of 128, 8 heads, 4 layers, and a total of 1M parameters.

Despite the Pixel-Space Generation showing positive results, it is important to remember that pixel-space generation was only performed on extremely simple data, and will not perform well on more realistic and diverse samples. Attempts to train on more advanced datasets either resulted in memory bandwidth concerns or extremely long training periods. Pixel-space generation only furthers the importance of image latent spaces for image generation

Latent Space Class-Conditional Generation

For class-based conditional generation the the transformer model size was increased to handle the more complicated datasets. The model configuration was an embedding size of 256, 8 attention heads, 4 activation layers, and in total, 3M parameters. Additionally, the team used the pre-trained VQ-GAN model with a downsampling factor of 4 for the Latent Space embeddings of each image.

The team performed class-based image generation on the Oxford Flower dataset, and the results can be viewed in Figure 4. For the flower dataset, the results contain the

distinctive shapes and colors of the various flower species. However, the results are quite blurry due to the downsampling effects and miss some finer details. For the CIFAR-10 dataset, some classes are easier to recognize than others (Figure 12). For example, automobiles and planes are very clear classes, while the dog and bird classes are not as distinct. Although CIFAR-10 consists of significantly fewer classes than the Oxford Flower dataset, the images are more diversified and less aligned. Therefore, it was a more difficult task for the transformer model to learn. Furthermore, on the even more diversified ImageNet dataset, the team was unable to get positive results, even after training a model with 20M parameters (Figure 15).

The team also observed that the conditional signal is only established during the late stage of training, causing the model to first learn the shape and color of the flowers, and then the connection between these concepts. Therefore, the model will first do unconditional generation, producing flowers with the wrong classes but focusing on image quality. Then, the model tends to generate flowers with the correct classes, along with improved quality results. This phenomenon can be noted in Figures 16, 17, and 18, which each display a training step. At the initial stage of training (2500 steps), the model gradually assembles patches into flowers, though with wrong classes; demonstrated by different colors and shapes in the same class. As the steps increase, the model makes fewer errors and generates flowers of the same class consistently.

Prompt Based Latent Conditional Generation

For Prompt Based Conditional Generation the same model in Section 5 was applied. The team additionally augmented the architecture with data argumentation layers and trained with a larger embedding size of 512, 5 layers, and 8 attention heads, resulting in 7M parameters. To learn the image caption text-embeddings the team applied an LSTM model to learn the correct embedding space for text, which is then provided to the transformer as additional input, along with the image codebooks. The team trained a three-layer character-level GRU LSTM with a hidden size of 3. The team performed character-level encoding instead of word-level because the corpus is small and short.

The results for text-to-image generation are displayed in Figures 5, 6, 21, 20, and 19. At first, the team was unable to output stable images. For example, when the prompt "The flower is red" was provided to the model, the output was a mix of red and white or even blue flowers. The team hypothesized that this is due to the text encoder's conditional signal being too weak, resulting in lackluster results. Therefore, the team emphasized the concept in the original prompt by repeating it several times. For example, the prompt would be modified to be "The flower is blue blue blue", forcing the model to recognize "blue" as a key phrase. When applying this prompt engineering practice, the team generated images with the correct colors (Figure 6). Additionally, applying the UNet VQ-VAE architecture, instead of the basic VQ-VAE improved the results substantially, as seen in Figures 5 and 6. The model that applied UNet VQ-VAE and data augmentation resulted in more detailed outputs. These results prove that the model can stably generate images with the correct conditions imposed.

Figure 5. Text-to-Image Generation - Basic VQ-VAE. Prompt: "The flower is blue"

Figure 6. Text-to-Image Generation - UNet VQ-VAE. Prompt: "The flower is blue blue blue"

Unfortunately, prompt engineering is only a trick, and due to the small size of the dataset, the team observed that the model does not correctly learn the cross-detail relationships

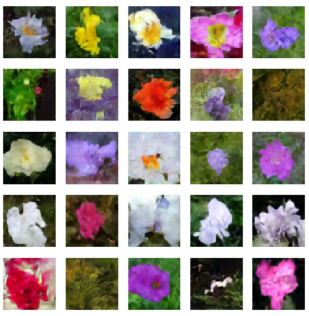

Figure 7. Text-to-Image Generation - Incorrect Output. Prompt: "The flower is red and blue"

between images. For example, the prompt "the flower is red and blue" will generate flowers will generate flowers that are either blue or red, but not a mix of them. (Figure 7. Similarly, the prompt "the flower is green or the flower is black" does not produce suitable results, because the Oxford Flower dataset does not have such colored flowers. However, while there are some limitations to this simple implementation of text-to-image generation, this still proves that it is feasible while sampling from the latent.

Similarly, the team also noticed the same phenomena that appeared in section 5. When reviewing the individual training steps, (300, 5400, 15000, and 98100) for text-conditional image generation, the model first learns the flower details, such as color and shape without regarding the initial text prompts, and then learns the textual conditions (Figures 22, 23, 24, and 25).

5.3. Sliding Window Results

In general, applying the Sliding Window Technique on the datasets resulted in overall worse results. For example, in Figure 26 the team applied the sliding window with a frame size of 13. The generated images are not very clear and are not distinguishable. Compared to the original images in Figure 27, the newly generated samples include face splicing. The team suspects this is due to the dataset only including low-resolution images. Applying a Sliding Window on low-resolution images causes the local features to forgo the global context. Applying this technique to a larger set of high-resolution images would likely produce better results, but the team was unable to confirm their hypothesis due to GPU limitations.

6. Discussion

6.1. Limitations

The main study limitation was the computational power available. The original Taming Transformers for High-Resolution Image Synthesis paper focused on unconditional image generation for high-resolution images, and as such their models reflected the requirements necessary for high-resolution datasets. This meant that the VQ-GAN Encoder and Generative-Transformer were required to scale due to the resolution increase. As the dimensions increase, the size of the sequence length increases quadratically due to the influx of pixel representations needed as codebook vectors. The research team was unable to experiment with high-resolution datasets due to the lack of powerful GPUs capable of supporting larger models. The team mostly experimented with lower-dimensional images since these were easier and faster to train while still maintaining the principles of generation. This also impacted the model evaluation, as the FID score calculations could not be provided with enough samples to be reliable and stable.

6.2. Implications

The main implication of this research study concludes that transformers are not limited to just natural language tasks, and can instead be applied to a wide range of problem statements. In this case, a Generative Transformer was augmented with conditional prompt information along with the encoded positional codebook indices. By learning to predict the next codebook index in the sequence, with contextual information from the image caption, the transformer was able to learn the relation between an image, and language. Overall, this shows that transformers are a general-purpose model architecture that can be applied outside of their intended initial scope by properly defining the problem statement around their unique set of problem-solving.

7. Conclusions and Future Work

In conclusion, this paper has displayed that for computer vision tasks sometimes taking a broader approach and applying non-traditional architectures to the problem statement may prove fruitful. In this case, the transformer architecture was capable of generating a large selection of images when sampling from an image latent space. By combining the power of VQ-VAE and Transformers, the team was capable of learning the different image datasets and generating never-before-seen samples. Furthermore, conditional generation via the LSTM text encoder allowed the models to interact with human speech and better understand human context.

7.1. Next Steps

High-Resolution Image Generation

As mentioned in Section 6.1, the main limitation of this study was the computational resources available; which severely limited the choice of training datasets. While the team was capable of getting admirable results on the low-resolution images, this architecture will be capable of being applied to everyday usage only after training with high-resolution images. To accomplish such a feat it would be necessary to scale up each sector of the model to accommodate for larger data. Working with a high-resolution dataset would require the computation power to support the increase in dimensions for the encoder and transformer.

Text-Embedding Improvements

As noted in section 5, one drawback of the model implementation was the need for prompt engineering. While the model is still capable of generating novel output, there are improvements to be made in the text-embedding architecture. Using a more sophisticated architecture for capturing the text embeddings can help to reduce the need for prompt engineering, and boost performance.

References

Chen, Mark, Radford, Alec, Child, Rewon, Wu, Jeffrey, Jun, Heewoo, Luan, David, and Sutskever, Ilya. Generative pretraining from pixels. *arXiv preprint arXiv:2001.08361*, 2020.

Chen, Xi, Kingma, Diederik P., Salimans, Tim, Duan, Yan, Dhariwal, Prafulla, Schulman, John, Sutskever, Ilya, and Abbeel, Pieter. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731*, 2016.

Esser, Patrick, Rombach, Robin, and Ommer, Björn. Taming transformers for high-resolution image synthesis. *arXiv preprint arXiv:2012.09841*, 2020.

Isola, Phillip, Zhu, Jun-Yan, Zhou, Tinghui, and Efros, Alexei A. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Oord, Aaron van den, Li, Yazhe, and Vinyals, Oriol. Neural discrete representation learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.

Parmar, Niki, Vaswani, Ashish, Uszkoreit, Jakob, Kaiser, Lukasz, Shazeer, Noam, Ku, Alexander, and Tran, Dustin. Image transformer. *arXiv preprint arXiv:1802.05751*, 2018.

Radford, Alec, Metz, Luke, and Chintala, Soumith. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations (ICLR)*, 2016.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III*, volume 18, pp. 234–241. Springer International Publishing, 2015.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, volume 30, 2017.

A. Appendix A

A.1. Datasets Applied

MNIST and Fashion MNIST

The Team started with the MNIST dataset, which consists of handwritten digits, and the Fashion MNIST dataset, which includes images of clothing items. Both datasets are well-known in the machine learning community for their simplicity and have been extensively used for benchmarking image classification models. Each dataset comprises grayscale images categorized into a small number of classes (10 classes for both datasets), making them ideal for initial experiments in class-based or label generation.

CIFAR Dataset

To broaden the scope, the team also incorporated the CIFAR dataset, specifically CIFAR-10, which includes 60,000 color images in 10 classes, with each class representing a different type of object such as birds, cars, and horses. This dataset provided a more challenging setting than MNIST and Fashion MNIST due to its color content and the greater diversity in image subjects. It allowed us to test and enhance the robustness of the image generation models in dealing with more complex and varied data.

Oxford 102 Flower Dataset

For more advanced experiments, particularly for text-to-image generation, the team utilized the Oxford 102 flower dataset. This dataset contains 102 different categories of flowers, with each category having several images, making it suitable for training the model to generate images from textual descriptions. The variability and complexity of the flower images compared to the simpler MNIST and CIFAR datasets provided a good stepping stone for testing the model's ability to handle more detailed and varied visual content based on textual prompts.

Final Notes

By employing these datasets, the team aimed to explore the capabilities of the proposed model across different complexities of image data and types of image generation tasks. This approach helped us gradually scale the experiments from basic label-based generation to more complex, descriptive text-based image synthesis.

A.2. Model Loss Functions

VQ-GAN Loss

$$\mathcal{L}_{\text{VQ}}(\{E,G,Z\}) = k\|x-\hat{x}\|^2 + k\sigma_g[E(x)] - \|z_q\|^2 + k\sigma_g[z_q] - \|E(x)\|^2. \quad (1)$$

$$\mathcal{L}_{\text{GAN}}(\{E,G,Z\},D) = [\log D(x) + \log(1-D(\hat{x}))] \quad (2)$$

$$\mathcal{L}_{\text{VQ-GAN}}(\{E,G,Z\},D) = \mathcal{L}_{\text{VQ}}(\{E,G,Z\}) + \lambda\mathcal{L}_{\text{GAN}}(\{E,G,Z\},D) \quad (3)$$

$$\lambda = \frac{\nabla_{\mathcal{G}}[\mathcal{L}_{\text{rec}}]}{\nabla_{\mathcal{G}}[\mathcal{L}_{\text{GAN}}] + \delta} \quad (4)$$

Transformer Loss

$$\mathcal{L}_{\text{Transformer}} = \mathbb{E}_{x\sim p(x)}[-\log(p(s))] \quad (5)$$

$$p(s_i|c) = Y_i \prod_i p(s_i|s_{i-1},c). \quad (6)$$

A.3. Supplemental Content

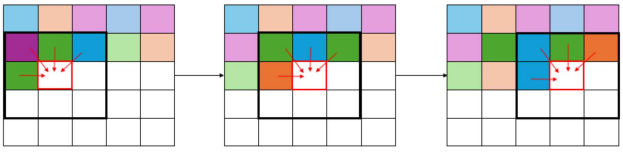

Figure 8. Sliding Window Technique

Figure 9. VQ-VAE Reconstruction Results. Top: Original, Bottom: Reconstruction

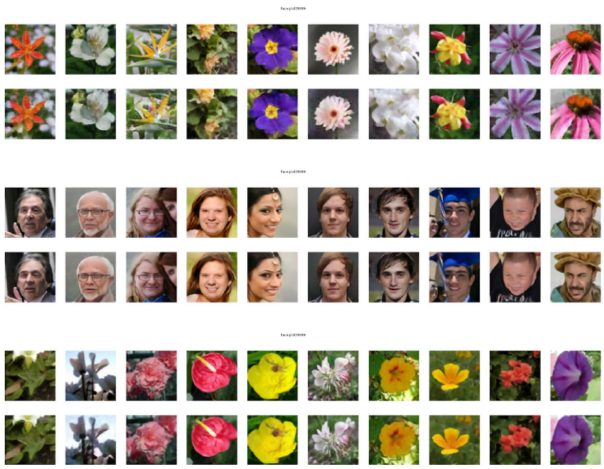

Figure 10. VQ-GAN Reconstruction Results. Top: Original, Bottom: Reconstruction

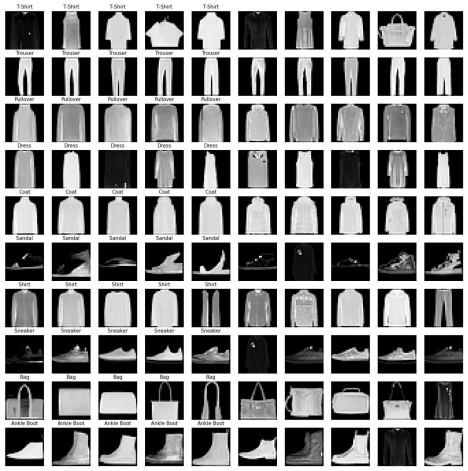

Figure 14. MNIST Fashion - Pixel Space Generation

Figure 11. VQ-GAN Finer Details Results

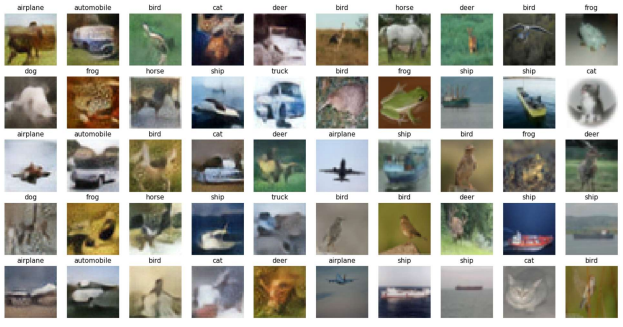

Figure 12. Class Conditional Generation on CIFAR-10. Left: Generated, Right: Original

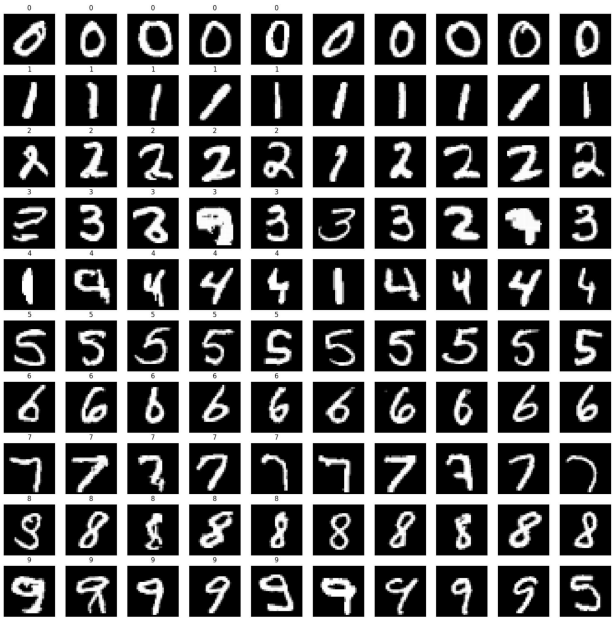

Figure 13. MNIST - Pixel Space Generation

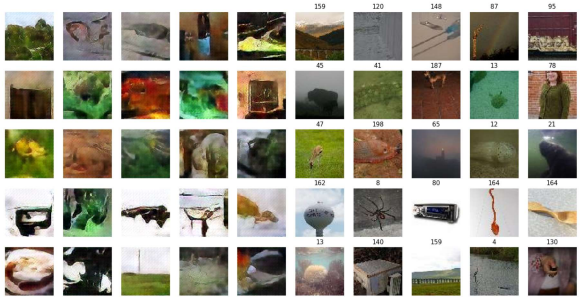

Figure 15. Class Conditional Generation on ImageNet. Left: Generated, Right: Original

Figure 16. Oxford Flower Class Conditional Generation - Step 0

Figure 17. Oxford Flower Class Conditional Generation - Step 2500

Figure 18. Oxford Flower Class Conditional Generation - Step 70000

Figure 19. Text-to-Image Generation - UNet VQ-VAE. Prompt: "The flower is yellow yellow yellow"

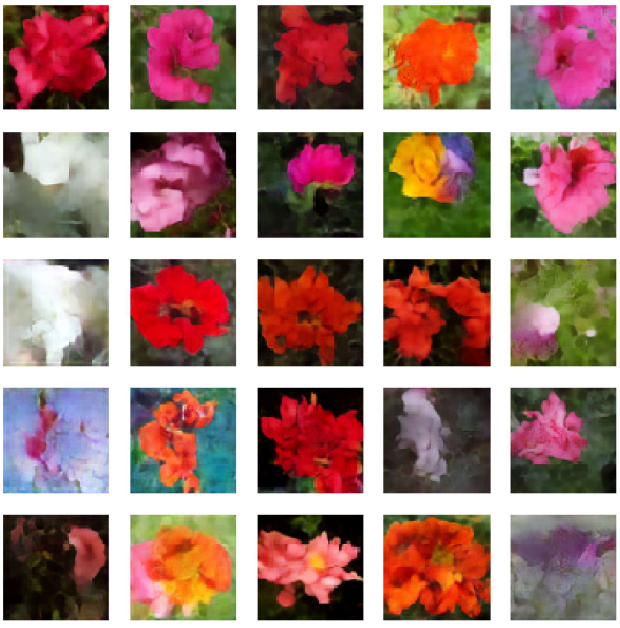

Figure 20. Text-to-Image Generation - Basic VQ-VAE. Prompt: "The flower is red red red red"

Figure 21. Text-to-Image Generation - Basic VQ-VAE. Prompt: "The flower is orange orange orange"

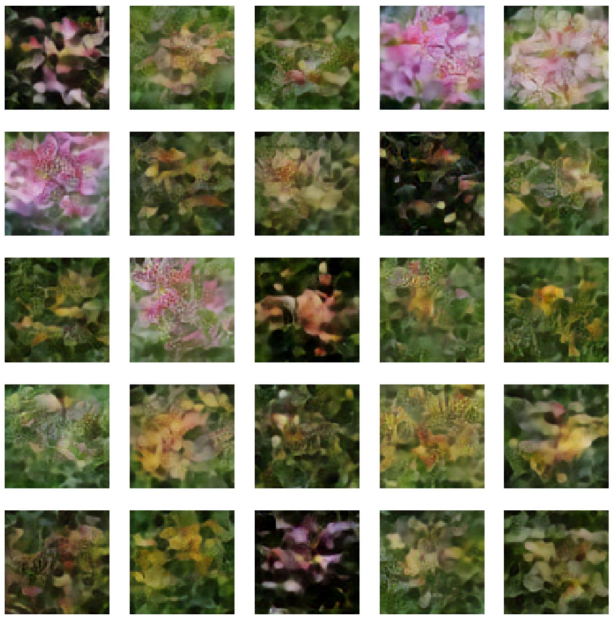

Figure 22. Oxford Flower Text Conditional Generation - Step 300

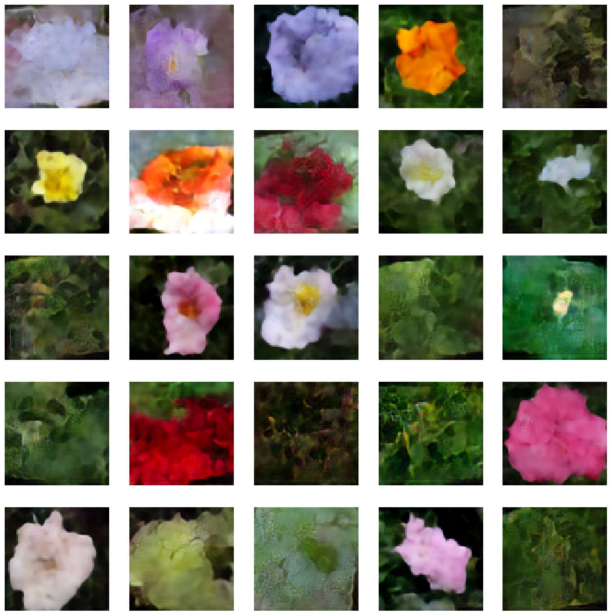

Figure 23. Oxford Flower Text Conditional Generation - Step 5400

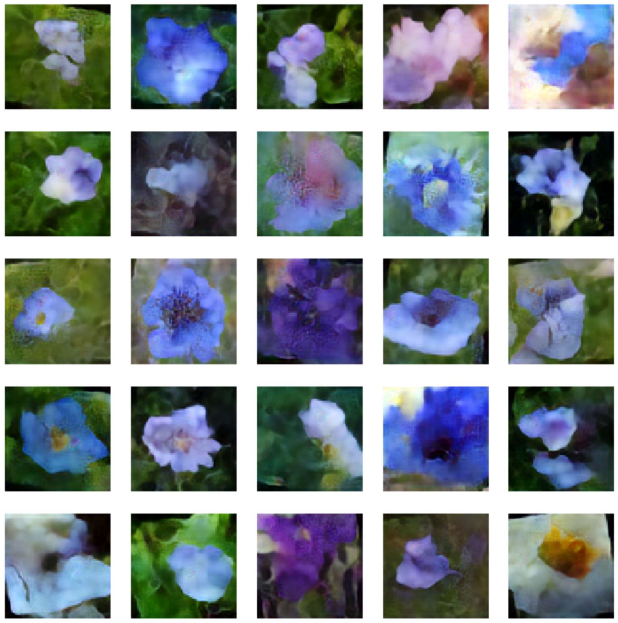

Figure 24. Oxford Flower Text Conditional Generation - Step 15000

Figure 25. Oxford Flower Text Conditional Generation - Step 98100

Figure 26. Face Dataset Sliding Window Results. Frame Size = 13

Figure 27. Original Face Dataset Images