Delft University of Technology, In Partial Fulfilment of the Requirements For the Bachelor of Computer Science and Engineering

Graphic Design with Machine Learning: Fabric Pattern Generation using a Variational Autoencoder

Just Wallage

June 24, 2021

Abstract

The goal of this research is to find out whether it is possible to use a specific type of machine learning algorithm to create new data that resembles data in a given dataset. The algorithm that is used is a variational autoencoder (VAE), this is a machine learning algorithm based around data compression and decompression. The dataset is a set of roughly 47,000 images of colorful fabric pattern designs created by a company called Vlisco. The working of a VAE is explained and implementations are discussed. Samples of generated images are shown and their quality is discussed. Eventually the results are compared to the results of a different research that focuses on the same problem but with a general adversarial network (GAN). If such algorithms work well they could potentially provide designers with new designs or inspiration quickly.

1 Introduction

This research aims to test existing machine learning methods on the generation of fabric pattern designs.

The Dutch company Vlisco [1] is specialized in creating fabrics with colorful patterns. These patterns are currently manually designed which is time consuming, designers need about one month to finish one design. See figure 1 for some examples of designs made by Vlisco. Fortunately Vlisco already has a large number of existing patterns that can be used as the input for a generative machine learning method which would make it possible to generate new pattern designs that are in the same style as the existing designs, but do not require the manual labour of designing them by hand. Such an algorithm would make it possible to create such designs much faster, on top of that it provides useful findings for generative methods on a type of data is not commonly used.

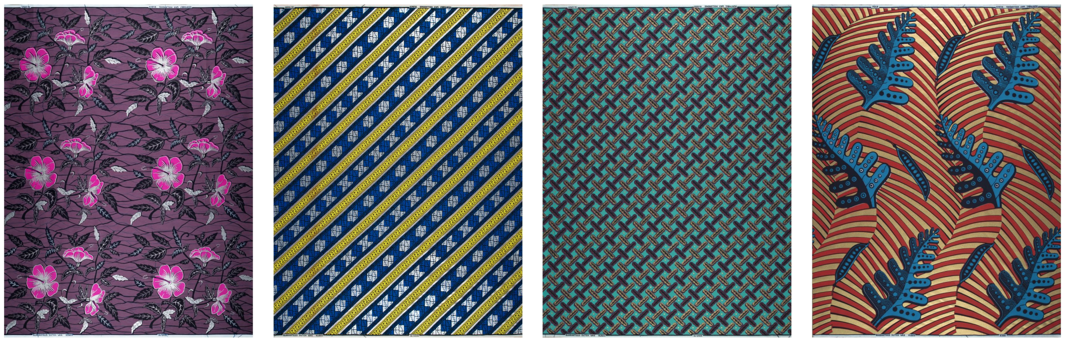

Figure 1: Some designs made by Vlisco [1]

This paper is focused on generating images based on existing images. This is done using a specific type of machine learning algorithm, namely the variational autoencoder (VAE). Variational autoencoders have been used in the past for image generation, such as in [5] and [7], however, the dataset used in this paper is much more diverse than the ones that are used in the referenced papers. On top of that the provided dataset is not categorised or labeled in any way.

This paper will start with the methodology (section 2), it provides a background on machine learning with images, a general explanation of VAEs and how VAEs can be used to generate images. Then section 3 will discuss more in detail how a VAE can be implemented for certain tasks and it explains which specific implementation is used for this research. Section 4 contains various results produced by the algorithm and discusses their quality. Section 5 goes in on an ethical question such a system might raise and the reproducibility of the experiments. Finally section 6 and 7 are the discussion and conclusion respectively, they focus on how successful the results are and possible future research on the topic.

The question that will be discussed in this paper is: "To which degree are variational autoencoders able to generate fabric pattern designs similar to the patterns provided by Vlisco?". To answer it, several existing versions of VAEs are compared, then some of those are implemented and tested. The results are subjectively discussed and compared to the original dataset. Finally the results of a similar research on the same problem but with a different type of machine learning (a general adversarial network) are compared to the results produced by the VAE in this research.

2 Methodology

This section describes the methodology of this research and provides background information on VAEs.

2.1 Artificial image generation

A common problem that arises when using ML with images is the data size images usually have. Most images that are used are at least several hundreds of pixels wide and tall, on top of that each pixel consists of either a single integer gray-scale value (typically between 0 and 255) or three such values for colored images, one for each color channel: red, green, blue. This poses a problem for neural networks because this increased size makes the neural networks larger and they therefore take more time to train. To reduce this issue, several techniques exist to reduce the size of an image without losing much of its important details.

Probably the most popular type of machine learning algorithm used for image generation is the general adversarial network (GAN) [2], but there exist many more. This paper focuses on variational autoencoders (VAE) [4], but there will be a similar research that will use a GAN to generate the same images, eventually the results of the two algorithms will be compared to find out which one works best.

2.2 Variational autoencoders

To understand VAEs we first need to understand how a regular autoencoder works. An autoencoder can be seen as a compression system that uses a neural network to first 'encode' some input data into a smaller size (i.e. less data), this is called the latent space, then it uses another neural network to 'decode' this latent space back into the input data, during the decoding phase the goal is to get the output as close as possible to the original input. Usually several neural network layers are used for the encoder and decoder, these are are called deep neural networks [3]. Autoencoders work well for data that is highly correlated with itself, for example images with many neighbouring pixels of the same color. Figure 2 shows a visual representation of an autoencoder, the green part in the center is the latent space.

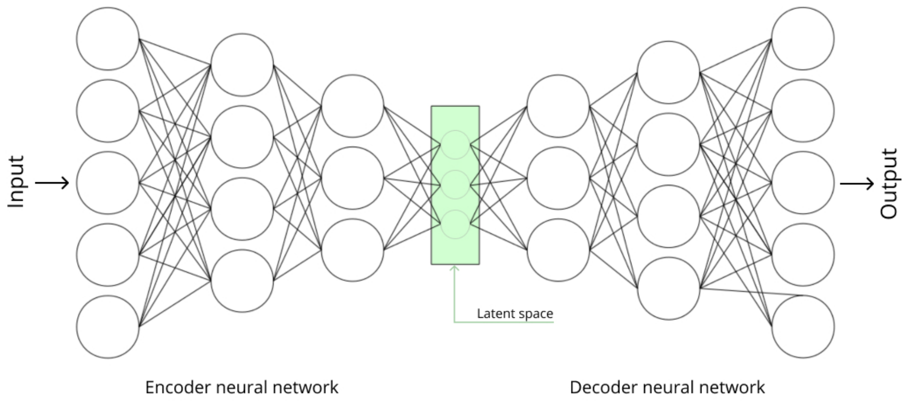

Figure 2: Visualisation of an autoencoder [11]

The difference between a regular autoencoder and a variational autoencoder is that while the latent space (encoded data) in a regular autoencoder is stored as a single vector of variables, a variational autoencoder uses two vectors for the latent space. These two vectors contain the mean value for each data-point and the standard deviation for each data-point (figure 3). To decode, the mean and standard deviation are used to sample a vector of data-points, this vector is then supplied to the decoder. This means that, while the latent space in a regular autoencoder is discrete, the latent space in a variational autoencoder is continuous.

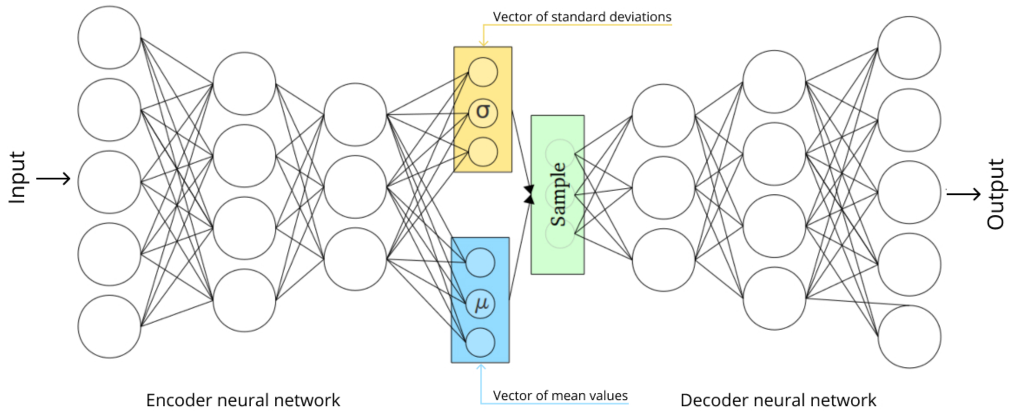

Figure 3: Visualisation of a variational autoencoder [12]

2.2.1 Image generation

An important advantage of the VAE when compared to a regular autoencoder is that it can be used to generate new data. Usually in machine learning, when creating new data, some form of random input is used. For an autoencoder this would mean providing a trained decoder with noise, however, such a model only produces new noise because as an observer it is unclear how the encoder encodes the input data. A variational autoencoder on the other hand can generate new images because it uses a normal distribution as the latent space. To generate new data using a variational autoencoder we can provide the decoder with a standard normal distribution and a well trained model should return new images that are similar to the training data.

2.3 Research methodology

The question is up to which level a VAE can generate designs similar to Vlisco's designs. To answer this, several steps are taken. First different implementations of VAEs are investigated. Some of these are implemented and tested with the Vlisco dataset. Once a promising implementation has been found further testing and tuning is done on this implementation. Finally the results are subjectively rated and they are compared to the results produced by the algorithm that is implemented by another research that focuses on the same problem but uses a GAN instead of a VAE [17].

3 Implementation

The scope of this project is to use one or more existing implementations of a VAE and test them on the Vlisco dataset. In the process of choosing a VAE several important aspects about VAEs need to be taken into account.

3.1 Variables of the model

3.1.1 Latent space

An important factor in a VAE is the size of the latent space. The latent space basically defines how much the decoder neural network has to do to get good results. A large latent space has as a result that the neural networks do not have to compress and decompress much and that makes it easier for the VAE to get good reconstructions of input images, but that also means that it is doing less work. On the other hand, a too small latent space might make it impossible for the decoder to reconstruct any image because it simply has too little information about the original input image. Therefore a good balance has to be found for the size of the latent space.

3.1.2 The dataset

The size and diversity of the training dataset can also make or break a model. Generally speaking the larger the dataset, the better the results. If there are not enough examples for the model to train, it will not be able to get good results. Another important aspect of the dataset is its diversity. A diverse dataset will likely perform poorly because the model cannot get used to a certain type of data. For example, when using a dataset that contains some gray-scale images, some color images and some black-and-white images the model will not understand what type of color mode it should use if the dataset is not sufficiently large enough to distinguish between the three.

3.1.3 Generation variables

When using a VAE for image generation we need a model that is tuned in the right way. As explained in section 3.1.1 the smaller the latent space, the more the neural networks have to do to get good results. That is also what is required when generating new images, as such, a small latent space will be used to train the model.

3.1.4 Training performance measurement

The metric that is used by the model to measure its performance is the mean squared error (MSE) between an input image and its reconstructed version. An MSE of zero means that two images are exactly the same. During training, the model considers a reconstruction with a low MSE as better and will tune its neural networks accordingly.

3.2 Experimental work

3.2.1 Vector quantized variational autoencoder

The vector quantized variational autoencoder (VQ-VAE) [5] seems to be a promising approach for the problem. As is in its name, it uses vector quantization: it quantizes the vectors in the latent space to the closest vector in a codebook [6]. The decoder uses this same codebook during the decoding phase to reconstruct the image. This technique allows for a greatly reduced latent space which speeds up the training process. Therefore the VQ-VAE is generally faster than a regular VAE which is why it can be used for higher resolution images more easily. There are several available implementations of this model and the one used in this research was written by rosinality [8].

3.2.2 Repetitive generation

An option used to get more interesting results is by reinserting a generated image back into the trained VAE model repeatedly. The VAE then encodes and decodes a generated image again and again which can add clearer characteristics of the trained model to the image. By creating an animated GIF image from the intermediate results you can get a feeling for how the VAE is slowly altering the image during each re-insertion. More on that in section 4.2. After some number of re-insertions the generated image reaches a certain state of clarity after which it does not evolve much more, but it still slowly changes its shape. When this state is reached one can keep on repeating until he gets a result that suits him.

4 Experimental Setup and Results

4.1 Grey-scale dataset

The first round of tests were done using a dataset consisting of 4000 grey-scale images provided by Vlisco. The size of this dataset can be considered fairly small as usually datasets of several tens of thousands of images are used for similar projects, for example [9] and [10]. As a result, the reconstructed images of by trained model are fairly blurry (see figure 4), a plot of the average MSE during training can be found in appendix section A.3. When using such a model to generate new images you get the results shown in figure 5.

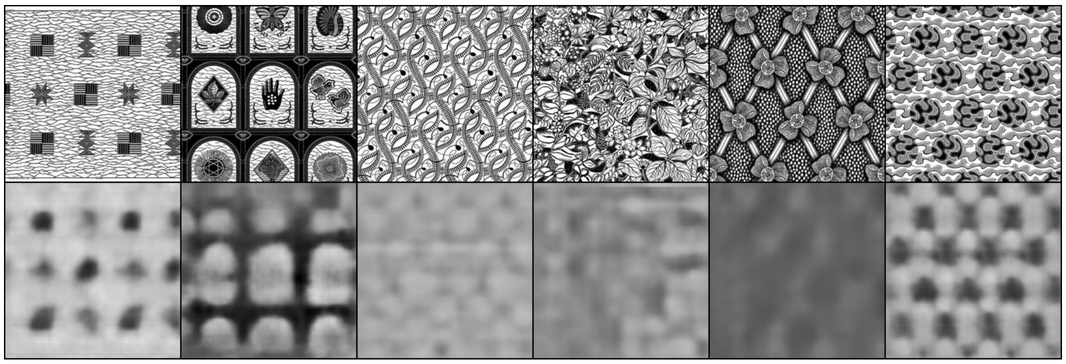

Figure 4: Some reconstructions of images from a grey-scale dataset that were encoded and decoded by a VQ-VAE model. The top row are the input images and the bottom row are the corresponding reconstructions after encoding and decoding. The latent space of this model is approximately 1000 times smaller than the size of the input data. The specific details of the input variables used in this model can be found in appendix section A.1.

As described in section 3.2.2, different results can be obtained by repeatedly reinserting a generated image back into the model. Figure 6 visualizes how an image evolves during

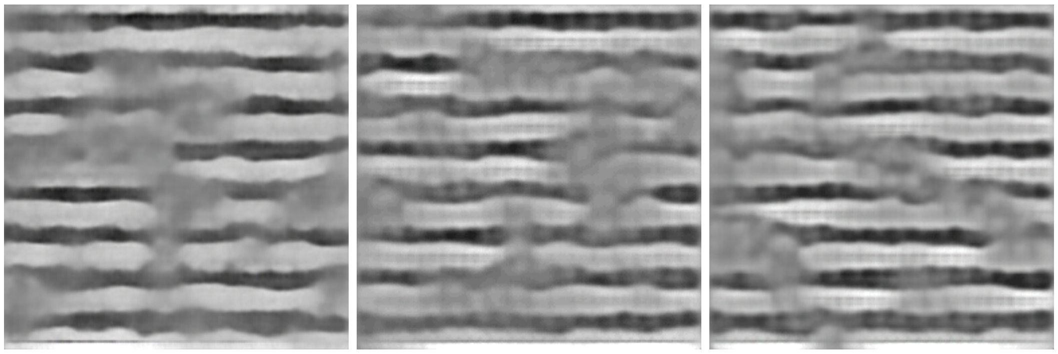

Figure 5: Some images generated by a VQ-VAE model that was trained on a grey-scale dataset. To generate the images the decoder of a trained VAE was provided with a vector created using a standard deviation for each value. The performance of this model for encoding and decoding original images can be viewed in figure 4.

the re-insertion process and how it slowly becomes a completely different image. During the first few rounds of reinserting the image loses a lot of detail, this is due to the fact that some information is lost when encoding and decoding. After more repetitions the differences per iteration begin to minimize, the cloudiness starts to disappear and the contrast becomes higher. These final steps make the images more pleasing to the human eye.

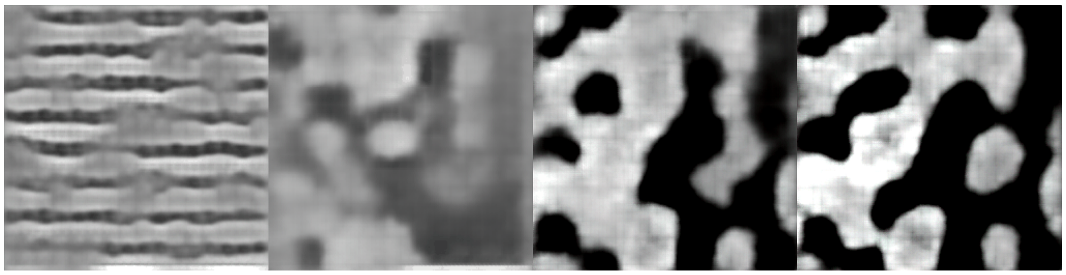

Figure 6: The result of reinserting (encoding and decoding) a generated image back into a trained VQ-VAE model repeatedly for 0, 20, 70 and 160 times respectively. See section 3.2.2 for an explanation of the process.

4.2 Colored dataset

The colored dataset provided by Vlisco consists of roughly 47.900 colored images. An important note is that most patterns are in the datasets multiple times in different color themes, for those images the pattern is exactly the same, but different colors are used. Figure 7 shows some reconstructions of colored images, similar to figure 4, but trained on images in the color dataset. These images clearly show the detail that is lost during encoding and decoding.

Randomly generated images by the color dataset are not impressive, see as an example first image of figure 8. These generated colored images show less detail than the generated grey images figure 5 showed. However, it is interesting to see how the colored images evolve when re-inserting a generated colored image (figure 8). One thing to note is that this model tends to generate a result with a lot of red and white colors eventually even though the initially generated input image contains little red.

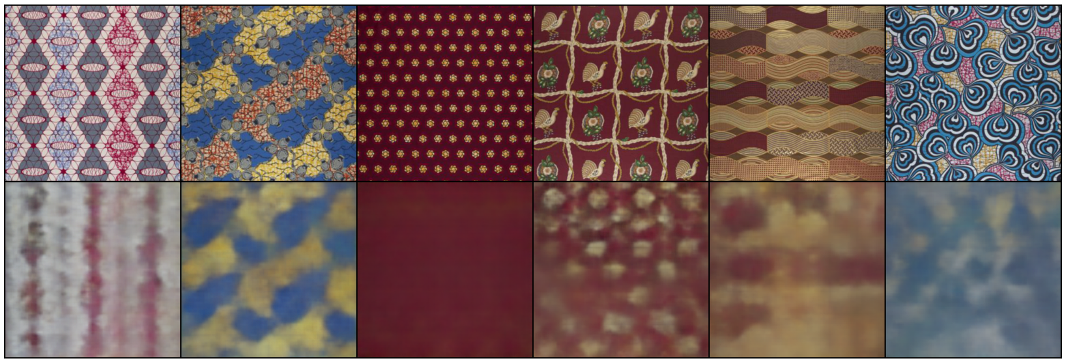

Figure 7: Some reconstructions of images from a colored dataset that were encoded and decoded by a VQ-VAE model. The top row are the input images and the bottom row are the corresponding reconstructions after encoding and decoding. The latent space of this model is approximately 1000 times smaller than the size of the input data. The specific details of the input variables used in this model can be found in appendix section A.1.

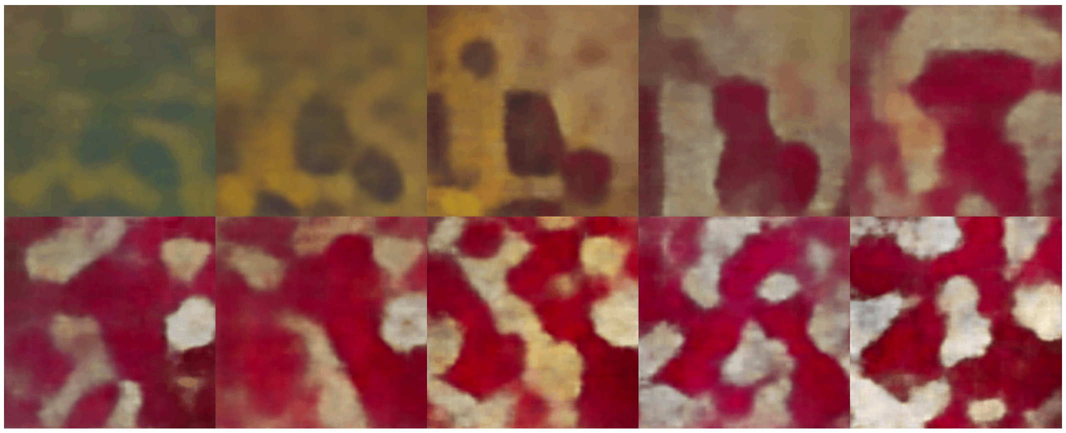

Figure 8: The result of reinserting (encoding and decoding) a generated image back into a trained VQ-VAE model repeatedly in steps of 20 re-insertions (i.e. 0, 20, ..., 160 and 180 times) respectively. See section 3.2.2 for an explanation of the process.

4.3 PixelSNAIL

In [5] first a vector quantized variational autoencoder (VQ-VAE, section 3.2.1) is trained, then a different type of machine learning algorithm is used to generate images, that algorithm is called PixelSNAIL [16]. This algorithm takes a trained VQ-VAE as input to train itself. It can then generate new images based on the trained VQ-VAE. This turns out to create more diverse results from the trained model when compared to the methods where the VQ-VAE was supplied with random data (sections 4.1 and 4.2). See figure 9 for the results that were produced by a trained PixelSNAIL model. It is clear that the images are much more colorful, which makes them more similar to the Vlisco dataset, however, the images are much more chaotic than the images by Vlisco and look more like coarsely colored noise with black borders around the colors. The chaotic nature is likely related to the diversity of the Vlisco dataset.

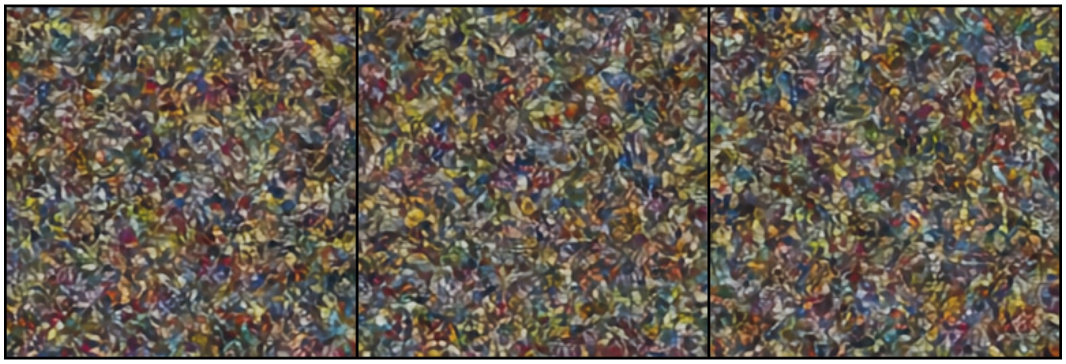

Figure 9: Generated images by a PixelSNAIL model that was trained on a trained VQ-VAE model, see section 4.3 for more details. For a full size version see section B.1.

4.4 Results overall

The current state of the produced results is still on another level of original designs than the images that are created by Vlisco. To answer the research question, it is safe to say that it is not yet possible to create designs that are comparable to Vlisco's designs. However, F. Schrama, a designer at Vlisco, was surprised by the generated results, especially the ones generated by L. Haarman's GAN [17], the animated GIF images from the repetitive re-insertions (section 3.2.2) also caught his attention as they visualize how the model alters images. Obviously the generated images themselves are not designs that will actually be turned into a fabric, however, Schrama mentioned that the results might be useful as an inspiration for the designers at Vlisco.

5 Responsible Research

5.1 Ethical aspect

A question that might come up while reading about artificial intelligence learning to create designs that were previously created by human designers is whether such an artificial intelligence can takeover the designers job. While this is currently not the case, it is hard to say how AI technology will develop in the future. Not only the increasing number of researches on the topic play a role, also the ever increasing performance of hardware has a large impact on the improvement of what AI is capable of doing. If AI machines are at one point in time able to deliver competitive designs, mankind could decide to have such work done by humans to save jobs, however, it will be hard to maintain such a decision on a global scale.

One could argue that government agencies should make rules concerning machines taking over human jobs, yet it is difficult to define clear rules to what types of machines would be allowed and what types would not be allowed. It is, however, likely that many human jobs are at risk of being taken over by machines because technology is improving rapidly.

5.2 Reproducibility

The methods used in this research are publicly available [5][8] and can be reproduced. The dataset provided by Vlisco, however, is not publicly available because of the competitive market position Vlisco aims to maintain. Obviously it is difficult to compare an implementation that was trained on different dataset, especially because of the unique and diverse nature of the Vlisco dataset. The exact input variables for the trained models can be found in appendix section A.1.

6 Discussion

6.1 Gray-scale and Vlisco

The grey-scale results may look interesting, they give an idea of what the VQ-VAE model creates, however, they are only marginally comparable to the images in the dataset that was provided by Vlisco. There are several possible reasons that could explain this. First of all, the size of the dataset that this model was trained on is of much smaller size than for example the datasets that were used in [5], the consequences of this are explained in section 3.1.2.

Another reason for the simple nature of the results could be the diversity of the Vlisco dataset. It is the models task to generalise all images, this can be really difficult or even impossible if each image is so vastly different and unique.

6.2 Color and Vlisco

The images that were generated by the color-image trained model are similar to the ones generated by the grey-scale model. This implies that the larger number of training images does not play a big role in the process. Likely, the dataset is still too diverse which makes it hard for the model to generalise and get an idea for what a generated image should look like, instead it creates a more general representation of the images which is not really comparable to the images in the dataset.

6.3 Comparing to GAN

As explained in section 2.3 the same problem is solved using another method, namely by using a GAN instead of a VAE [17]. In general GANs have been known to perform well on image generation and this is can also be concluded by this research. Appendix section C.1 shows some designs that were created by s GAN trained on the Vlisco dataset. The images are much more diverse and detailed than the ones generated by the VAE, however, they are still not close to the original designs created by Vlisco.

6.4 Resources

When training a model with images, a lot of computational power is required. For this research two high performance computing sources were used: the TU Delft high performance cluster (HPC) and Google Colab [15]. Due to limitations as a student on the HPC it could only be used for short tests with the gray-scale dataset. Since the color dataset contains much more images it was trained on Colab. Colab offers more freedom in data storage and computational power, however, it is not an infinite source because users get kicked from the service if they use too much resources within a short time. If the computational resources were less limited it would have been possible to train the model for a longer time period which could possible result in better results.

6.5 Remaining problems

A problem that the current model faces is that it does not recognise the real world objects that often occur in the designs. This is a major flaw because it is unable to create new designs with real world objects which is what makes them stand out. Instead the model can only try to create a general understanding of the entire dataset as a whole which leads to more abstract results. If somehow the model is able to understand and create real world objects the model could possibly produce more interesting results, for example by combing it with an image recognition model such as ImageNet [13] and an object to image generation model such as DALL-E [14].

Another issue most machine learning algorithms face is training time. While this is not really an issue per se, if training was faster it would be possible to try different implementations and settings much quicker which would increase the progress of such researches. For example the PixelSNAIL model took several days to train before it could generate the images shown in figure 9. To decrease training time one could either try to optimize the algorithm to make it run more efficiently or use more powerful hardware, but this is financially not efficient.

7 Conclusions and Future Work

It became clear that the current state of the model used in this research is not capable of generating designs that could compete with the ones designed by the designers at Vlisco, the detail of those designs is still much higher. However, designers may be able to use the generated designs as a source of inspiration for their next project. Once trained, a model can deliver new designs rapidly, that is one of the main advantages computers have over humans. Designers can quickly get inspiration from as many randomly generated designs as they please.

To get better results with a variational autoencoder, a dataset that is less diverse is likely necessary. The model seems unable to generalize on the used datasets and therefore generates either too simple images or images that look like coarse colorful noise with black borders around each color. Other implementations of the VAE may be able to create better results, yet it is unlikely that at this point in time VAEs are able to generate images indistinguishable to Vlisco's images. In the future, further research may introduce new technologies such as better hardware or new innovations in the field of variational autoencoders.

References

[1] Vlisco. https://www.vlisco.com/.(accessed 12.06.2021).

[2] Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). Generative adversarial networks. arXiv preprint arXiv:1406.2661.

[3] Bengio, Y., & LeCun, Y. (2007). Scaling learning algorithms towards AI. Large- scale kernel machines, 34(5), 1-41.

[4] Kingma, D. P., & Welling, M. (2014). Auto-encoding variational bayes.Conference proceedings: papers accepted to the International Conference on Learning Representations (ICLR) 2014. Paper presented at the 2nd International Conference on Learning Representations (ICLR2014)

[5] Razavi, A., Oord, A. v. d., & Vinyals, O. (2019). Generating diverse high-fidelity images with vq-vae-2. arXiv preprint arXiv:1906.00446.

[6] Flanagan, J. K., Morrell, D. R., Frost, R. L., Read, C. J., & Nelson, B. E. (1989). Vector quantization codebook generation using simulated annealing. Paper presented at the International Conference on Acoustics, Speech, and Signal Processing.

[7] Kingma, D. P., & Welling, M. (2013). Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114

[8] Seonghyeon, K. (2020). rosinality/vq-vae-2-pytorch. https://github.com/rosinality/vq-vae-2-pytorch.(accessed 12.06.2021).

[9] The CIFAR-10 dataset. https://www.cs.toronto.edu/ kriz/cifar.html.(accessed 12.06.2021).

[10] ImageNet. https://www.image-net.org/.(accessed 12.06.2021).

[11] Dertat, A. (2017). Applied deep learning-part 3: Autoencoders. Towards Data Science, 3. https://towardsdatascience.com/applied-deep-learning-part-3-autoencoders-1c083af4d798. (accessed 12.06.2021).

[12] Shafkat, I. (2018). Intuitively Understanding Variational Autoencoders. Towards Data Science. https://towardsdatascience.com/intuitively-understanding-variational-autoencoders-1bfe67eb5daf. (accessed 12.06.2021).

[13] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. Advances in neural information processing systems, 25, 1097- 1105

[14] Ramesh, A., Pavlov, M., Goh, G., Gray, S., Chen, M., Child, R., . . . Agarwal, S. (2021). Dall-E: Creating images from text. https://openai.com/blog/dall-e/.(accessed 12.06.2021).

[15] Google Colab. https://colab.research.google.com/.(accessed 12.06. 2021).

[16] Chen, X., Mishra, N., Rohaninejad, M., & Abbeel, P. (2018). Pixelsnail: An improved autoregressive generative model. Paper presented at the International Conference on Machine Learning

[17] Haarman, L. (2021). Investigating the performance of Generative AdversarialNetworks on Fabric Pattern Generation

[18] Maximilian Seitzer. (2020). pytorch-fid: FID Score for PyTorch. https://github.com/mseitzer/pytorch-fid.

A Appendix

A.1 Input variables for latent space 1/1000 size

These are the variables that were used for the VQ-VAE2 implementation with a latent space of 1000 times the size of the input images, table 1.

Input size	256 x 256
Latent layers	4 x 4, 8 x 8
Commitment loss coefficient	0.25
Batch size	16
Hidden units	128
Residual units	64
Layers	6
Codebook size	512
Codebook dimension	64
Encoder cone filter size	3
Upsampling cone filter size	4

Table 1: The variables used for the VQ-VAE2 implementation with a latent space of approximately 1000 times smaller than the original input. All variables are 8 bit data points.

A.2 Input variables original algorithm

These are the variables that were used for the original VQ-VAE2 implementation, table 2.

Input size	256 x 256
Latent layers	32 x 32, 64 x 64
Commitment loss coefficient	0.25
Batch size	16
Hidden units	128
Residual units	64
Layers	2
Codebook size	512
Codebook dimension	64
Encoder cone filter size	3
Upsampling cone filter size	4

Table 2: Original input variables for the VQ-VAE2 implementation. All variables are 8 bit data points.

A.3

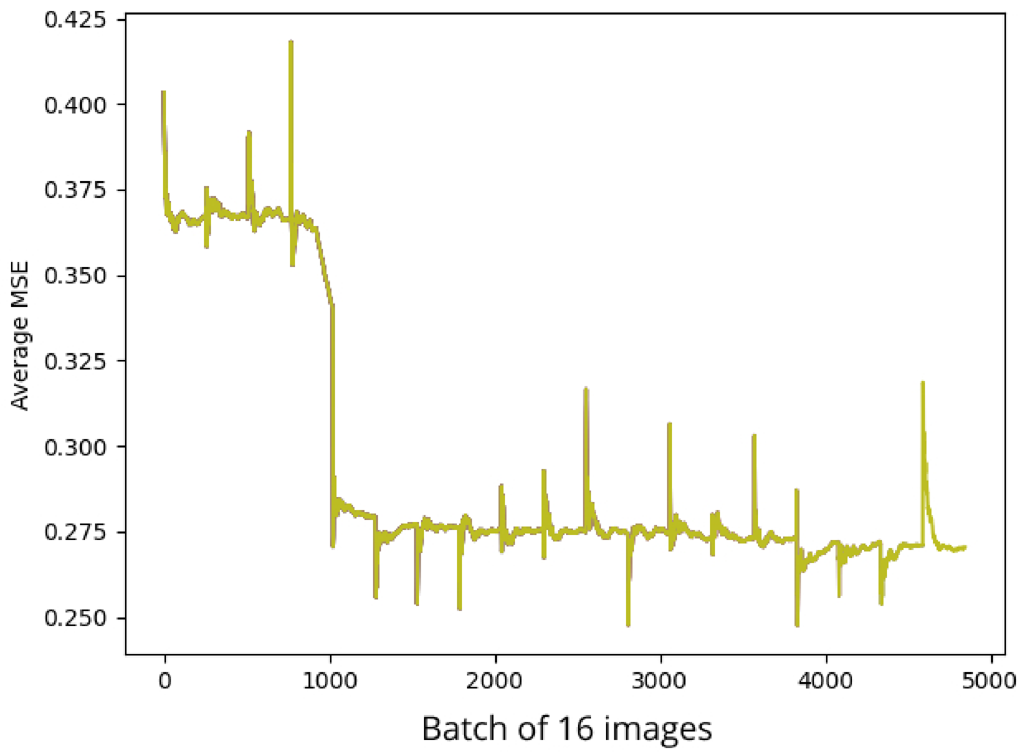

Figure 10: Plot of the average MSE between the input and reconstructed images per batch of 16 images. The latent space of this model was approximately 1000 times smaller than the input size.

B

B.1

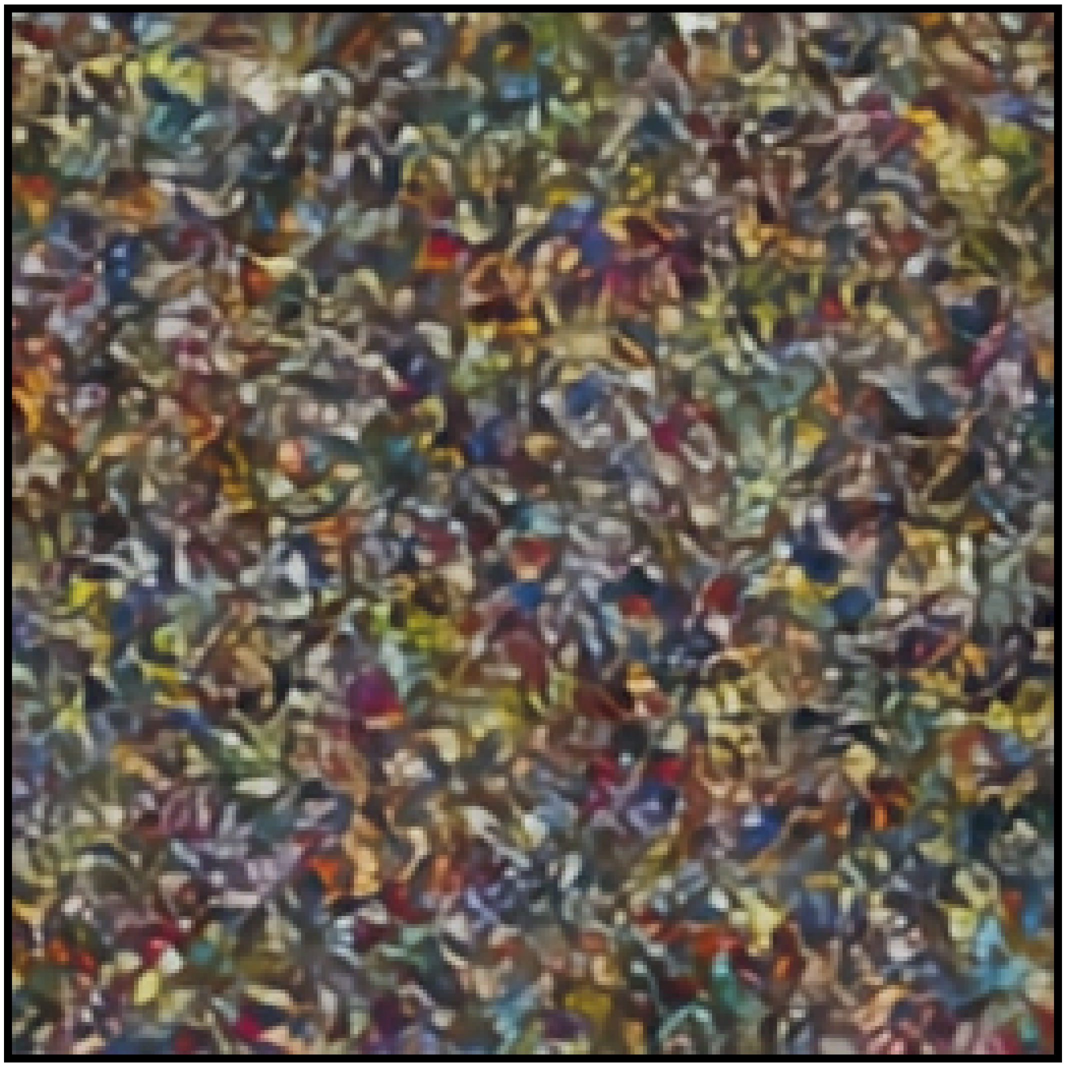

Figure 11: A full size 256x256 PixelSNAIL generated image

C

C.1

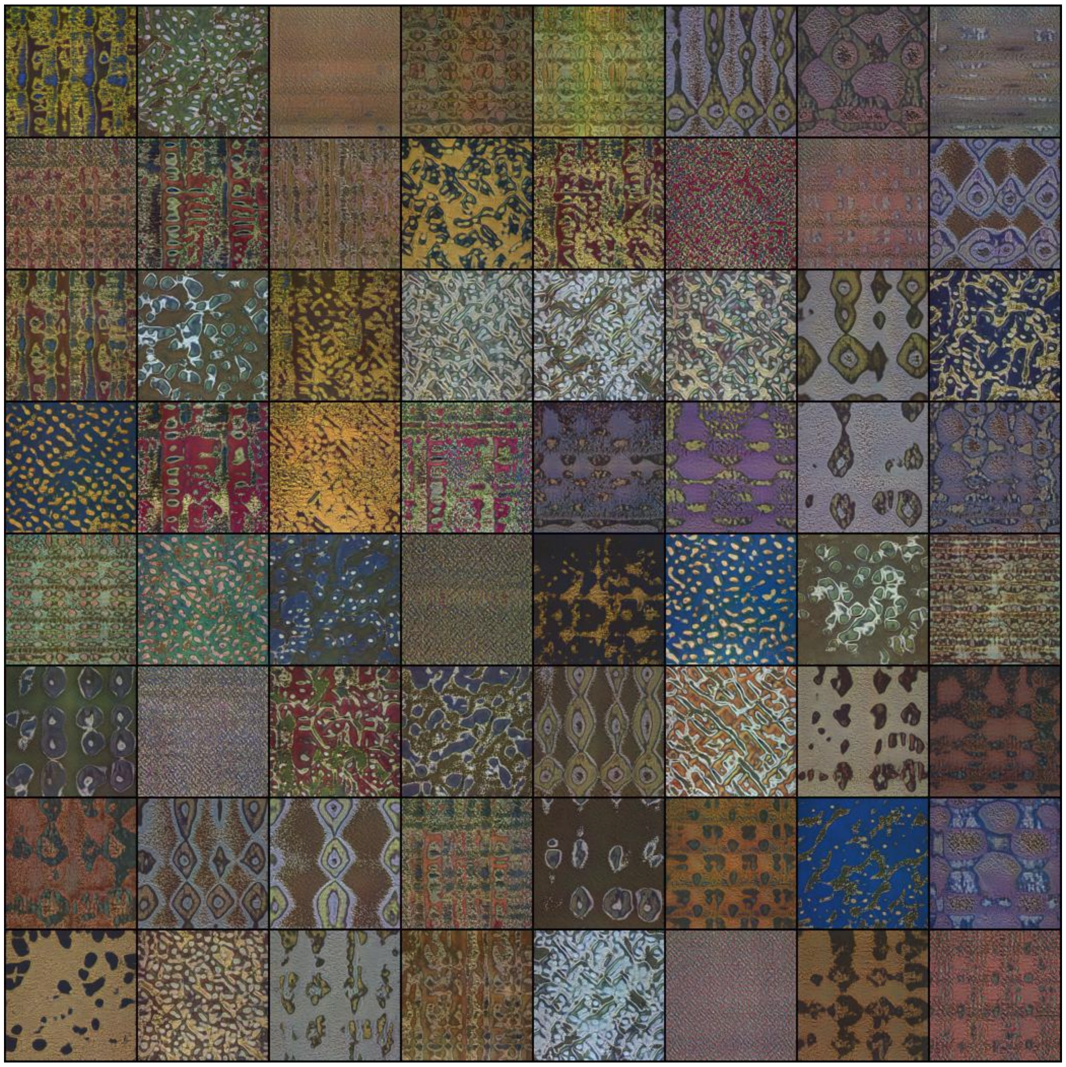

Figure 12: Some images generated by StyleGAN2 that was implemented to create designs similar to Vlisco's.

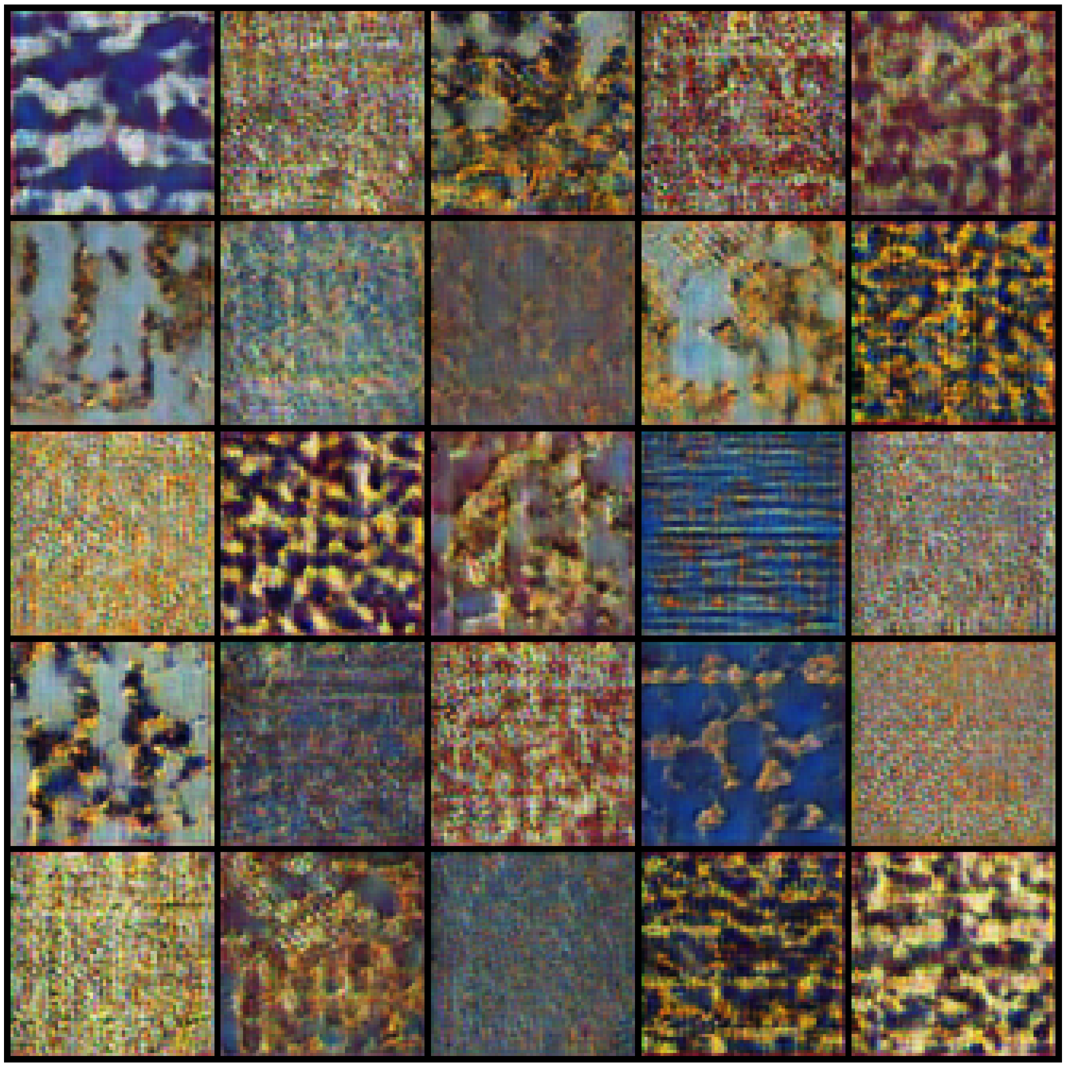

Figure 13: Some images generated by RelastivisticGAN that was implemented to create designs similar to Vlisco's.