RE-IMPLEMENTATION OF SWAPPING AUTOENCODER FOR DEEP IMAGE MANIPULATION

Callum Newlands, Jacob Scott, Muhammed Qaid
Electronics and Computer Science
University of Southampton, UK
{cn2g18, js11g18, mq1g18}@soton.ac.uk

ABSTRACT

Whilst deep generative models have been successfully utilised to synthesise realistic images, using these models to instead perform controlled manipulating of existing images has remained difficult. The "Swapping Autoencoder" was proposed as a solution to this problem by Park et al. (2020). This work re-implements the presented architecture and experiments, finding that the observed performance and functionality remains close to those presented in the original paper.

1 INTRODUCTION

The purpose of this paper is to review the reproducibility of the paper "Swapping Autoencoder for Deep Image Manipulation", (Park et al., 2020). In the paper, the authors propose a "Swapping Autoencoder" model for controllable manipulation of existing, real images. Their model achieves impressive results, successfully able to swap the structure and texture of images to create realistic hybrids and to apply manageable global and local edits by utilising latent code vector arithmetic. This reproducibility report aims to re-implement[1] their model and experiments in an effort to examine to what extent their results can be reproduced, and at what cost (e.g. computationally).

The key idea behind the "Swapping Autoencoder" architecture is to encode the input images into separate structure and texture codes. These can then be swapped between images or manipulated before being used to by the decoder to re-create the image. The model is trained by swapping these codes between pairs of images and then ensuring that the hybrid images are realistic and visually similar with a novel "co-occurrence patch discriminator". The idea of swapping of codes during training is not original; however, unlike previous approaches, the "Swapping Autoencoder" is fully unsupervised. Similarly, the decomposition of the image into texture and structure allows for more control in image manipulation and results in higher-quality hybrid images.

1.1 SCOPE OF RE-IMPLEMENTATION (METHODOLOGY)

Our initial goal was to successfully build and train our implementation of the model (as described by the paper). With our implementation of the model, we attempted to reproduce the results of the paper demonstrating the model's main image manipulation functionality—primarily focusing on the model's capabilities rather than comparisons to other baseline models. In particular, we aimed to reproduced the experiments shown in **Figures 3, 4 and 7** of the original paper, as these showcase the model's ability to separate texture and structure components, reconstruct realistic images, and facilitate controlled image manipulations.

2 IMPLEMENTATION DETAILS

Our encoder E, generator G, and patch discriminator D_{patch} follow the same framework of the original paper. The softplus version of the standard non-saturating GAN loss (Goodfellow et al., 2014; Wang et al., 2019) was used due to the outputs of the discriminators lying outside the range $[0, 1]$. The auto-encoder loss is given by:

$$\mathcal{L}_{\text{auto-encoder}} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{GAN}} + \mathcal{L}_{\text{cooccur}} \tag{1}$$

[1] Repository for our re-implementation:
https://github.com/COMP6248-Reproducability-Challenge/Swapping-Autoencoder-for-Deep-Image-Manipulation-Reproduction

where

$$\begin{aligned}
\mathcal{L}_{\text{rec}}(E, G) &= \|x - G(E_{\text{rec}}(x))\|_1, \\
\mathcal{L}_{\text{GAN}}(E, G, D) &= \mathbb{E}_{x\sim X}[0.5 \cdot \text{softplus}(-D(G(E_{\text{rec}}(x)))) + 0.5 \cdot \text{softplus}(-D(G(E_{\text{swap}}(x))))], \\
\mathcal{L}_{\text{cooccur}}(E, G, D) &= \mathbb{E}_{x\sim X}[\text{softplus}(-D_{\text{patch}}(G(E(x_{\text{patch}})), x_{\text{patch}}))].
\end{aligned}$$

The discriminator is in place to enforce realistic images, and the patch discriminator to ensure swapped images look similar to the true images. The combined discriminator loss is given by:

$$\mathcal{L}_{\text{discriminator}} = \mathcal{L}_{\text{patch\_real}} + \mathcal{L}_{\text{patch\_swap}} + \mathcal{L}_{\text{GAN\_real}} + \mathcal{L}_{\text{GAN\_fake}} \tag{2}$$

where

$$\begin{aligned}
\mathcal{L}_{\text{patch_real}}(D_{\text{patch}}) &= \mathbb{E}_{x\sim X}[\text{softplus}(-D_{\text{patch}}(x_{\text{patch}}, x_{\text{patch}}))], \\
\mathcal{L}_{\text{patch_swap}}(D_{\text{patch}}) &= \mathbb{E}_{x\sim X}[\text{softplus}(D_{\text{patch}}(x_{\text{swap_patch}}, x_{\text{patch}}))], \\
\mathcal{L}_{\text{GAN_real}}(D) &= \mathbb{E}_{x\sim X}[\text{softplus}(-D(x))], \\
\mathcal{L}_{\text{GAN_fake}}(E, G, D) &= \mathbb{E}_{x\sim X}[0.5 \cdot \text{softplus}(D(G(E_{\text{rec}}(x)))) + 0.5 \cdot \text{softplus}(D(G(E_{\text{swap}}(x))))].
\end{aligned}$$

The autoencoder and discriminators were trained in alternating (adversarial) steps, with lazy-R1 regularisation being carried out on the discriminators as in StyleGAN 2 (Karras et al., 2019).

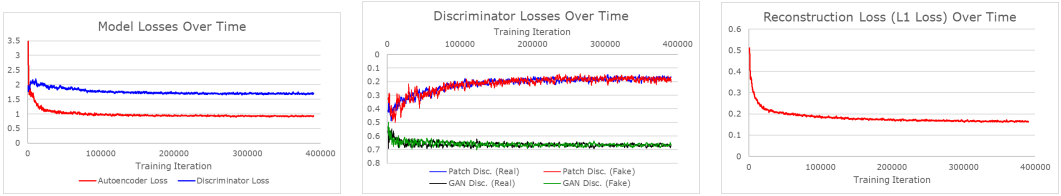
The re-implemented model was trained for the same number of iterations as the original work ($2.5 \times 10^7/\texttt{batch\_size}$) on the LSUN Church dataset (Yu et al., 2015). For wall-time-saving purposes, the images were downscaled to 64×64px and a batch size of 64 was used. The model was trained on the Iridis Compute Cluster[2] across four Nvidia RTX 8000 GPU cards with 48GB GDDR6 and training took around 5 days. The model was saved every 50 iterations to minimise the impact of any possible service outage and to log training losses and evaluation metrics.

One difference in model architecture from the original paper was the kernel size of the final convolutional layer in the texture channel of the encoder. Due to the reduced image resolution, and the use of zero padding, a 1×1 kernel size was necessary in place of the 3×3 used in the original work.

All training and model code was re-implemented from scratch, aside from the data loading code[3] and the components of the StyleGAN2[4] architecture re-used in the original paper—as these were not part of the original authors' contributions, and thus lie outside the scope of this report.

3 Results and Analysis

3.1 Reconstruction and Training Performance



(a) Red: autoencoder loss. Blue: discriminator loss

(b) Red and Blue: patch disc. Green and Black: GAN disc.

(c) L1 reconstruction loss

Figure 1: Plots of the model losses (a), discriminator accuracies on real and fake images (b) and L1 reconstruction loss (c) over the training iterations.

Figure 1 shows the model performance over time during training. The performance quickly improved over the first 100,000 iterations, and then performance increases slowed considerably. This

[2]The authors acknowledge the use of the IRIDIS High Performance Computing Facility, and associated support services at the University of Southampton, in the completion of this work.

[3]https://github.com/taesungp/swapping-autoencoder-pytorch

[4]https://github.com/rosinality/stylegan2-pytorch

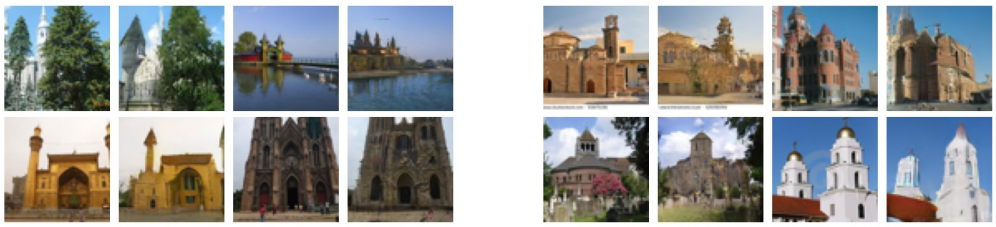


Figure 2: **Examples of images reconstructed by our trained model.** For each pair of images: left = input, right = reconstruction. Left: The trained model can accurately preserve colour and shape of the image, as well as difficult features such as trees, water, windows and doorways. Right: Some hand-selected examples to show the worst performance of the model—whilst the overall shape of the building and colour palette has been preserved, some finer details in the textures have been lost, in particular the positions of windows and roof tiles.

could be due to the increased batch size (64 compared to the 16), or due to hyperparameters (such as the learning rate) needing to be changed as a result of the downscaled images. Both the autoencoder and discriminator losses decreased at roughly equal rates, and both stabilised—rather then diverging—which suggests that the adversarial training process was successful.

Figure 2 shows some example of reconstructed images by the final trained model. The trained model can be seen to accurately preserve the structure and shapes in the image to a highly accurate level—particularly the main outline and the positions of towers and doors. As well as this, the colours and overall style of the images are reconstructed almost exactly. For some inputs, the model performs less well—particularly regarding the fine details of textures such as windows and roof tiles—but the overall structure and style is still highly recognisable. This reconstruction loss is likely due to the architecture and training hyperparameters being tuned to large scale images, and a lower reconstruction loss could most likely be obtained with parameter tuning or training on higher resolution images.

3.2 Image Manipulation

Figure 3 shows the results of texture modifications with image encoding swapping. The resulting images are realistic and mostly accurately match the structure and texture of the relevant input

Figure 3: **Examples of image structure-texture swapping modifications.** Each image shows the result of combining the texture encoding from the image in top row and the structure encoding from the image in the left row and then decoding the result. The trained model has clearly learned to disentangle the structure and texture components, and can produce realistic images from swapped encodings.

images closely. This shows that the trained model architecture facilitates the separation of structure and texture features, as well as the fact that the model can reconstruct realistic images from swapped encodings. Some visual artefacts are present—such as the addition of domed rooves in the first column of the right image—but the images produced are still realistic and these could likely be trained out by adjusting the training hyperparameters for the smaller-sized images.

As well as direct swapping manipulations, the trained model supports manipulation vectors in the same manner as the original work. Figure 4 shows the interpolated results of applying a calculated 'night-time' vector to the texture channels of three images. Under this process, the model is still able to produce realistic images which have the same structure as the input images, whilst matching the desired style (visually closer to day time or night time).

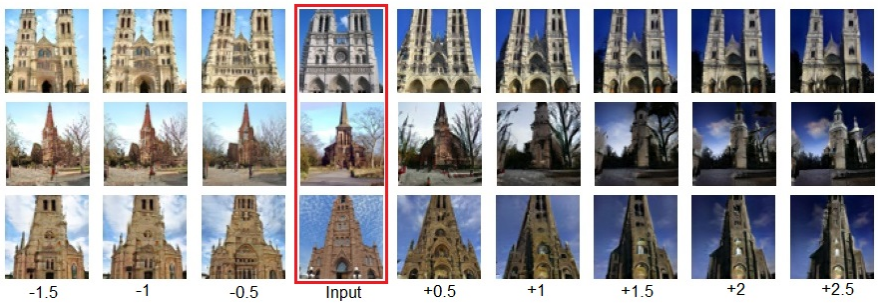


Figure 4: **Examples of continuous interpolation using a style manipulation vector.** A manipulation vector for 'night-time' is found by taking the mean difference between the texture encodings for 20 hand-selected photos of night and day. This vector is then added to the encoded texture encoding for the input image with some scalar gain (below). The model can be seen to not only change the appearance of the sky, but also add or remove shadowing to various parts of the image as necessary—and not just applying a general darkening/colour filter.

4 Conclusion

We have re-implemented the "Swapping Autoencoder" from Park et al. (2020) and trained our re-implememented model on the LSUN Church dataset at a downscaled resolution of 64×64px. Overall the trained model is able to accurately split and reconstruct images with similar results to the original paper, and supports the same image manipulation processes proposed in the original work: structure-texture swapping and style manipulation vectors. Our model performance is slightly lower than the originally presented work, however—taking into account the flattening of the training loss plots—this is likely due to the reduced input image resolution requiring hyperparameter adjustments for optimal training. One aspect of the original work that was outside the scope of this re-implementation, but may benefit from a extension reproducibility study, is interactive image editing through a user interface.

References

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of StyleGAN. *CoRR*, abs/1912.04958, 2019.

Taesung Park, Jun-Yan Zhu, Oliver Wang, Jingwan Lu, Eli Shechtman, Alexei A. Efros, and Richard Zhang. Swapping autoencoder for deep image manipulation. In *Advances in Neural Information Processing Systems*, 2020.

Wei Wang, Yuan Sun, and Saman Halgamuge. Improving MMD-GAN training with repulsive loss function. In *International Conference on Learning Representations*, 2019.

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop. *CoRR*, abs/1506.03365, 2015.