CAGAN: Text-To-Image Generation with Combined Attention Generative Adversarial Networks

Master's Thesis of

Henning Schulze

at the Department of Informatics
Interactive Systems Lab Institute for Anthropomatics and Robotics
Karlsruhe Institute of Technology (KIT)

Supervisors: Prof. Dr. Alexander H. Waibel
M.Sc. Dogucan Yaman

01. March 2020 – 28. August 2020

Karlsruher Institut für Technologie
Fakultät für Informatik
Postfach 6980
76128 Karlsruhe

Hiermit versichere ich, dass ich diese Arbeit selbständig verfasst und keine anderen, als die angegebenen Quellen und Hilfsmittel benutzt, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht und die Satzung des Karlsruher Instituts für Technologie zur Sicherung guter wissenschaftlicher Praxis in der jeweils gültigen Fassung beachtet habe.

Karlsruhe, 28. August 2020

. . . . . .

(Henning Schulze)

Abstract

Generating images according to natural language descriptions is a challenging task. In this thesis, we propose the Combined Attention Generative Adversarial Network (CAGAN) to generate photo-realistic images according to textual descriptions; and we critically discuss evaluation metrics for text-to-image generation. By combining squeeze-and-excitation attention with word attention and applying spectral normalisation, a GAN stabilising technique, our proposed CAGAN improves the state of the art on the inception score for the Caltech-UCSD Birds 200 dataset while generating (mostly) realistic images and showing a reasonable text-image correlation. We demonstrate that judging a model by a single evaluation metric can be misleading by searching for opposing responses of evaluation metrics and by developing an additional model which outperforms the state of the art on the inception score while generating unrealistic images through feature repetition. Furthermore, we demonstrate that a second popular evaluation metric, the Fréchet inception distance, is calculated differently by multiple papers, thereby inhibiting a fair model comparison. Our thesis stresses the need for the use of more than one evaluation metric; a unified evaluation approach in the field of text-to-image generation; and ideally an evaluation metric offering a fair model comparison.

Zusammenfassung

Das Generieren von Bildern aus natürlichen Sprachbeschreibungen ist eine anspruchsvolle Aufgabe. In dieser Thesis, schlagen wir das Combined Attention Generative Adversarial Network (CAGAN) vor, um fotorealistische Bilder nach textuellen Beschreibungen zu generieren; und wir führen eine kritische Diskussion über Evaluationsmetriken für Text-zu-Bildgenerierung. Indem wir squeeze-and-excitation Aufmerksamkeit mit Wortaufmerksamkeit kombinieren und spektrale Normalisierung anwenden, verbessert unser vorgeschlagenes CAGAN den state of the art des inception scores auf dem Caltech-UCSD Birds 200 Datensatz, während es (meist) realistische Bilder generiert und eine vernünftige Text-Bild-Korrelation aufweist. Wir demonstrieren, dass die Bewertung eines Models aufgrund einer einzigen Evaluationsmetrik irreführend sein kann, indem wir nach entgegengesetzten Reaktionen von Evaluationsmetriken suchen und indem wir ein weiteres Modell entwickeln, welches den state of the art auf dem incetpion score verbessert, während es durch Merkmalwiederholungen unrealistische Bilder erzeugt. Des Weiteren demonstrieren wir, dass eine zweite populäre Evaluationsmetrik, die Fréchet inception distance, von mehreren Papern unterschiedlich berechnet wird. Dadurch wird ein fairer Modellvergleich beeinträchtigt. Unsere Thesis bekräftigt die Notwendigkeit mehr als eine Evaluationsmetrik zu benutzen; eines einheitlichen Vorgehens bei der Evaluation im Felde der Text-zu-Bildgenerierung; und idealerweise einer Evaluationsmetrik, welche einen fairen Modellvergleich gewährleistet.

Acknowledgements

The research conducted towards this thesis was part of the Continuous Learning in International Collaborative Studies (CLICS) exchange program and was funded by the Germany Academic Exchange Service (DAAD).

Contents

List of Figures

List of Tables

1. Introduction

Generating images according to natural language descriptions spans a wide range of difficulty, from generating synthetic images to simple and highly complex real-world images. It has tremendous applications such as photo-editing, computer-aided design, and may be used to reduce the complexity of or even replace rendering engines, not having to simulate complex light transport, surface geometry, shading, etc. [50] Furthermore, good generative models involve learning new representations. These are useful for a variety of tasks, for example classification, clustering, or supporting transfer among tasks.

Generating images highly related to the meanings embedded in a natural language description is a challenging task due to the gap between text modality and image modality.

There has been exciting recent progress in the field using numerous techniques and different inputs [39] [11] [22] [32] [9] [30] [58] [56] [57][59] yielding impressive results on limited domains. A majority of approaches are based on Generative Adversarial Networks (GANs) [19]. A GAN is composed of two networks: a generator and a discriminator which are jointly trained with a competing goal in an adversarial manner. The discriminator evaluates the difference between the real and the generated data distribution, thereby avoiding to directly compare these distributions. This advantage led to GANs demonstrating impressive performance in generative tasks.

Zhang et al. introduced Stacked Generative Adversarial Networks [84] which consist of two GANs generating images in a low-to-high resolution fashion. The second generator receives the image encoding of the first generator and the text embedding as input to correct defects and generate higher resolution images. Recently, a number of techniques have been proposed improving upon Stacked GANs [86] [82] [52] [8] [38] [10] [53] [81] [85].

Xu et al. [81] improve on Stacked GANs by introducing AttnGAN which utilises a novel loss function and fine-grained word attention. We build on their approach, investigating the introduction of several known attention models into the existing model and proposing methods of combing attention. Furthermore, we discuss the impact of attention on different parts of the network.

Given that there has been a lot of recent research and progress in the field of self-attention and that the existing model contains a complex word-attention model, we focus on the introduction of several known self-attention mechanism. These include global, local, spatially-aware, and light-weight approaches. We stabilise the training behaviour of the GAN by applying spectral normalisation [44] in the discriminator. Additionally, we repurpose attention mechanisms to obtain sentence attention.

Having a number of attention maps from different models leads to the question of how to combine them. We experiment with different views of attention, translating attention maps to a common view, and combining attention maps. These encompass common network structures, such as convolutions, normalisations, and activations as well as tensor operations.

We evaluate our proposal using several of the most popular evaluation metrics for generative image modelling, including the Inception Score (IS) [61] and the Fréchet Inception Distance (FID) [24]. By combining squeeze-and-excitation attention with word attention and applying spectral normalisation, a GAN stabilising technique, our proposed Combined Attention Generative Adversarial Network (CAGAN) boosts the IS of our baseline by 9.6% ± 2.4% from 4.36 ± 0.04 to 4.78 ± 0.06 and improves the state of the art by 0.6% ± 2.8% from 4.75 ± 0.07 to 4.78 ± 0.06 on the CUB dataset [74].

Furthermore, our proposed CAGAN boosts the FID of our baseline by 10.0% from 47.76 to 42.98. A comparison to the FIDs of the state of the art is futile, because several papers report no FID score and those that do report vastly different FID scores on the CUB dataset for the same baseline suggesting the use of different FID implementations.

A subjective, qualitative visual analysis illustrates that our proposed CAGAN generates images of similar quality to the AttnGAN and shows reasonable generalisation abilities. In addition, we investigate different views of attention and methods of combining attention.

We critically discuss evaluation metrics for text-to-image generation and for evaluating GANs. We demonstrate the importance of reporting more than on evaluation metric by analysing the anti-correlation of our evaluation metrics by searching for opposing responses, i.e., occurrences of improving on one metric while deteriorating on another metric; and by developing a model boosting our baseline on one specific evaluation metric, the IS, by 13.8% ± 2.2% from 4.36 ± 0.04 to 4.96 ± 0.05 while generating completely unrealistic images through feature repetitions and having a major negative impact on the FID of our baseline of 27.8% from 47.76 to 61.06.

We revisit a critical discussion of the inception score and the overall suitability of all our measures in the context of text-to-image generation. Moreover, we show that several recent papers [86] [8] [10] report vastly different FID scores on the CUB dataset for the same baseline suggesting the use of different FID implementations. Our work stresses the need for the use of more than one evaluation metric; a unified evaluation approach in the field of text-to-image generation; and ideally an evaluation metric offering a fair model comparison.

Lastly, we observe mode collapse when applying attention in the discriminator; examples of internal anti-correlation; and different behaviour of evaluation metrics.

2. Background

This chapter provides a brief introduction into the underlying methods used in the following chapters. Section 2.1 is an excursion to the biological motivation of neural networks. Section 2.2 introduces artificial neural networks and covers common activation functions (Subsection 2.2.1), feed-forward neural networks (Subsection 2.2.2), common loss functions (Subsection 2.2.3), gradient descent (Subsection 2.2.4), backpropagation (Subsection 2.2.5), and popular improvements such as batch normalisation (Subsection 2.2.6) and residual connections (Subsection 2.2.7).

Section 2.3 introduces Convolutional Neural Networks (CNNs), their basic building blocks convolutional layers (Subsection 2.3.1) and pooling layers (Subsection 2.3.2), and gated linear units (Subsection 2.3.3), an activation function designed for CNNs. Section 2.4 introduces recurrent neural networks, their accompanying vanishing gradient problem (Subsection 2.4.1), and one of its solutions: long short term memory (Subsection 2.4.2). Section 2.5 introduces generative adversarial networks and discusses several training instabilities (Subsection 2.5.1) such as convergence, vanishing gradients, and mode collapse. Lastly, Section 2.6 provides a brief introduction of autoencoders and denoising autoencoders (Subsection 2.6.1).

2.1. Biological Background

Artificial neural networks were inspired by biological brains and are used for a variety of machine learning tasks. Biological neurons consist of the soma (cell body), axons, dendrites, and synapses. A spike or action potential is a short (1 ms) and sudden increase in voltage created by the soma. The axon is the signal carrier of a spike. Incoming signals alter the voltage of a neuron. The synapses work as signal pre-processors and alter the membrane voltage positively or negatively when receiving a spike. This makes them crucial for learning and adaption. If the membrane voltage of a neuron reaches above a threshold value the neuron sends out a spike. After the spike the neuron enters a short refractory period (10 ms). The refractory period is a short moment of rest in which the neuron can not send out another spike.

The spikes of biological neurons are all similar. However, the postsynaptic potentials vary in size, i.e., the effect of incoming spikes varies. Individual neurons send out erratic spike trains (sequences of spikes) which alter dramatically in frequency over a short period of time. Therefore, neurons have to use spatial and temporal information of incoming spike patterns to encode their messages to other neurons. In addition to electrical as-

pects, biological neurons also have chemical aspects in form of transmitters enhancing or diminishing the effects of spikes. [73] [20] [42]

2.2. Artificial Neural Networks (ANNs)

Artificial Neural Networks (ANNs) use a strongly simplified version of biological neurons ignoring chemical and temporal aspects: a single neuron receives input x over connections; the input is multiplied by weights w; then a bias b is added; and the result is put in an activation function φ yielding the output y:

$$y = \varphi(w^T x + b) = \varphi(\sum_i w_i \cdot x_i + b) \ . \tag{2.1}$$

2.2.1. Activation Functions

The activation function is typically non-linear, examples are the sigmoid function:

$$\sigma(x) = \frac{1}{1 + e^{-x}} \ , \tag{2.2}$$

the ReLU, and the LeakyReLU [41]:

$$\sigma(x) = x^+ = max(0, x) \tag{2.3}$$

$$\sigma(x) = \begin{cases} x, & \text{if } x \geq 0 \\ \alpha \cdot x, & \text{otherwise} \end{cases} \ . \tag{2.4}$$

2.2.2. Feed-Forward Neural Networks (FFNN)

An ANN is defined by a structure x and internal parameters θ:

$$y = ANN(x, \theta) \ . \tag{2.5}$$

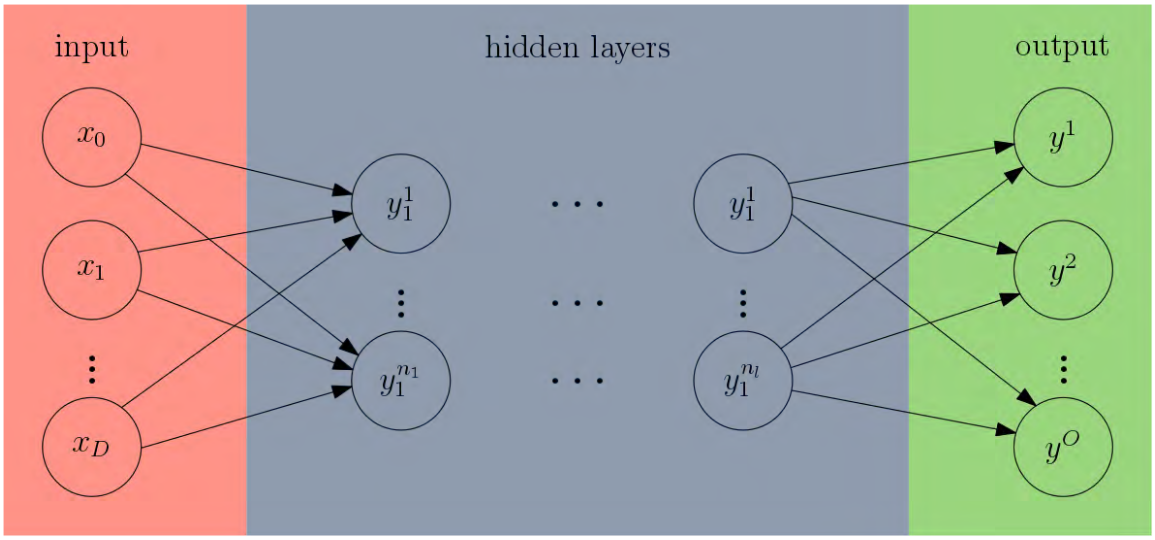

Figure 2.1.: Feed-forward neural network with D inputs, l hidden layers, each of size n_k, and O outputs. [1]

A common structure is the Feed-Forward Neural Network (FFNN). A FFNN groups neurons into layers and restricts connections, such that no cycle is allowed. Often, each layer only receives the output of the previous layer as input, see Figure 2.1. We assume this case in the following. Each layer k can be interpreted as function f_k that receives an input x_k and yields an output y_k; we denote W_k as the weight matrix between layer k and layer $k-1$ and b_k as the biases for the neurons in layer k:

$$y_k = f_k(x_k) = f_k(y_{k-1}) = \varphi(W_k y_{k-1} + b_k) \ . \tag{2.6}$$

Thus, the output of the network is:

$$y = f_n \circ f_{n-1} \circ ... \circ f_1(x) \ . \tag{2.7}$$

2.2.3. Loss Functions

The parameters W_k and b_k are learned and not designed. This requires a loss function. A loss function compares the output of the network y to a desired output y'. Therefore, it is a supervised learning problem. A typical loss function is the Mean Squared-Error (MSE/L2):

$$L(y, y') = \frac{1}{n} \cdot \sum_{i=1}^{n} (y_i - y_i')^2 \ . \tag{2.8}$$

[1] Figure was created by the author.

2.2.4. Simple/Stochastic Gradient Descent

To learn the parameters, the contribution of each parameter to the error is computed. The contribution of a weight w_{ij} is defined as the partial derivative of the loss function of the weight w_{ij}:

$$\frac{\partial L(y, y')}{\partial w_{ij}} \ . \tag{2.9}$$

When using simple gradient descent as optimizer, the weight w_{ij} is updated according to:

$$w'_{ij} = w_{ij} + \eta \cdot \frac{\partial L(y, y')}{\partial w_{ij}} \ , \tag{2.10}$$

where η is the learning rate. Considering all training samples for each weight update is costly and the gradient descent may get stuck in local minima. To approach these problems, stochastic/batch gradient descent (SGD) considers only a single/fixed number of training samples per weight update. This results in more weight updates and the ability to escape local minima. However, the weight updates are less meaningful and a smaller learning rate is recommended.

2.2.5. Backpropagation

Backpropagation provides an efficient way to compute the gradients of a loss function with respect to the individual weights:

$$\frac{\partial L}{\partial w_{ij}} = \frac{\partial L}{\partial y} \frac{\partial y}{\partial w_{ij}} \ . \tag{2.11}$$

With the definition of Equation 2.6 and Equation 2.7 we can invoke the chain rule:

$$\frac{\partial y}{\partial w_{ij}} = \frac{\partial f_n}{\partial y_{n-1}} \frac{\partial f_{n-1}}{\partial y_{n-2}} \cdots \frac{\partial f_{j+1}}{\partial y_j} \frac{\partial f_j}{\partial w_{ij}} \ . \tag{2.12}$$

Backpropagation computes the gradients layer by layer, starting at the back. Each partial derivative $\frac{\partial f_j}{\partial y_{j-1}}$ is only computed once and reused for the previous layer, thus making it an efficient algorithm.

2.2.6. Batch Normalisation

Batch normalisation [29] is a technique that reduces overfitting, allows better generalisation, stabilises training, and increases convergence velocity by allowing higher learning rates.

Batch normalisation introduces a new layer before the activation function which fixes the means and variances. Ideally, the normalisation is conducted over all training examples. In practice, the normalisation is conducted over a mini-batch allowing it to be used in

conjunction with SGD. Batch normalisation normalises the output of the previous layer by subtracting the batch mean and then dividing by the batch standard deviation. The resulting output has zero mean and unit variance.

Batch normalisation was originally designed to reduce the internal covariance shift, i.e., the shift around the mean. However, recent work [62] has shown that batch normalisation does little in that regard, rather its effectiveness results from a significant smoothing of the optimisation landscape. Thereby, it introduces a more predictive and stable behaviour of gradients.

2.2.7. Residual Connections

He et al. [23] introduced residual connections which advanced the state of the art for multiple image-related tasks. A residual connection directly adds the output of a layer to a later layer, thereby skipping the residual block (the layers in between). Usually, the residual block consists of a few layers. The only restriction of the residual block is that its output size must match its input size. The principle is depicted in Figure 2.2.

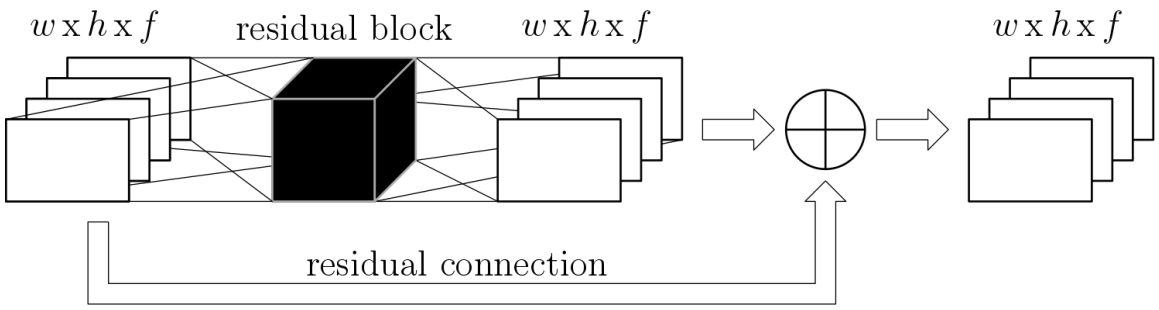

Figure 2.2.: Residual connection for an arbitrary residual block with matching output and input size. [2]

Residual connections provide the network with more paths: each residual block has two paths. Therefore, a network with n residual blocks has 2^n paths. One advantage of having more paths is that gradients no longer vanish, because there is a direct path for the gradient. Veit et al. [71] show this behaviour by demonstrating that most of the gradients during training of a deep network originate from short paths. This allowed the training of networks that are two magnitudes deeper than previous models with up to thousands of layers.

Moreover, Veit et al. show that entire layers of a trained residual NN can be removed while maintaining comparable performance. Similar procedure in common NNs decreases the performance, because common NNs only provide a single path for the gradient. Removing a layer compromises that path. In a residual NN, however, the residual connection remains after removing a layer providing an uncompromised path.

[2] Figure was created by the author.

2.3. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) [37], interpretable as an extension of Time-Delay Neural Networks (TDNNs) [75], introduce two new layers: the convolution layer and the pooling layer. CNNs are commonly used for image-related tasks due to their ability to deal with the high dimensionality of images. For example, a simple 128x128 RGB image as input translates to $128 \cdot 128 \cdot 3 = 49152$ parameters. Fully connecting them with a very simple layer of a thousand neurons results in $49152 \cdot 1000 \sim 49.2$ million parameters to be learned.

2.3.1. Convolutional Layer

The convolutional layer addresses the dimensionality issue via weight sharing. Instead of having each neuron depend on all neurons of the previous layer, each neuron simply depends on its mirror neuron, which is the neuron of the previous layer occupying the same spatial position, and a small neighbourhood around it. Furthermore, the weights among any neuron and its mirror neuron plus neighbourhood are identical, i.e., shared. This ensures that the same features are being detected, regardless of their spatial position, and it greatly reduces the number of parameters.

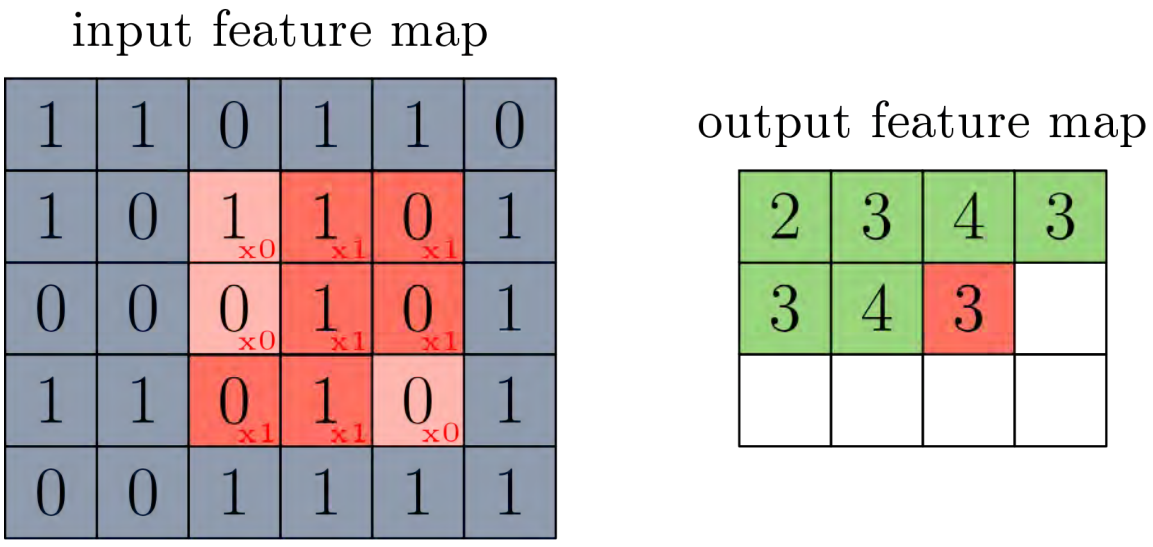

Figure 2.3.: The 7th step of a convolution with a filter-kernel of size 3x3. [3]

A convolutional layer performs a convolution with n filter kernels, each of size kxk. Figure 2.3 shows a step of a 2D-convolution with a single feature map and a kernel of size 3x3. For a 2D-convolution the input is required to be two-dimensional or higher. Typically, the input consists of multiple 2D feature maps, for example, one feature map for each colour-channel of an RGB image. Each kernel operates on all input feature maps and

[3] Figure was created by the author.

produces a single feature map as output. Thus, the number of output feature-maps equals the number of kernels.

The input feature maps are being overlayed with the kernel which is then moved step by step. In each step the overlayed elements are multiplied by the learned kernel weights and then summed up yielding a single value in the output feature map. To preserve the spatial size of the input feature map, the input may be padded. The step-size may also be adjusted to reduce the spatial size of the output feature map.

2.3.2. Pooling Layer

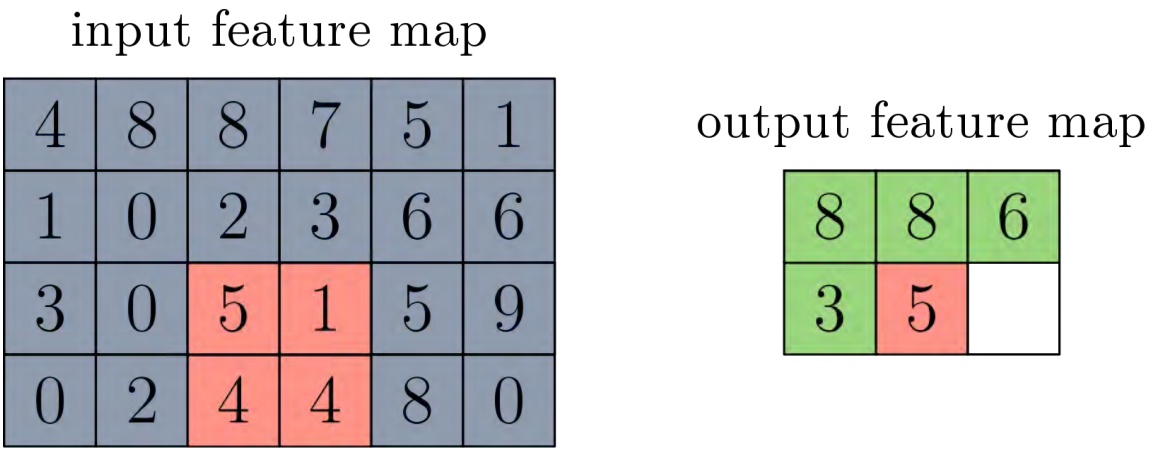

Figure 2.4.: The 5th step of a max-pooling with a kernel of size $2x2$. [4]

The purpose of the pooling layer is to reduce the spatial size of the representation. Ideally, the pooling layer also prevents overfitting and removes noise while keeping the relevant information. The pooling layer combines multiple values of a neighbourhood into a single value according to a fixed rule. Examples for the fixed rule are: max-pooling, where the maximum of the values is the output, and average-pooling, where the mean of the values is the output. Figure 2.4 is an example of a $2x2$ max-pooling.

[4]Figure was created by the author.

2.3.3. Gated Linear Unit (GLU)

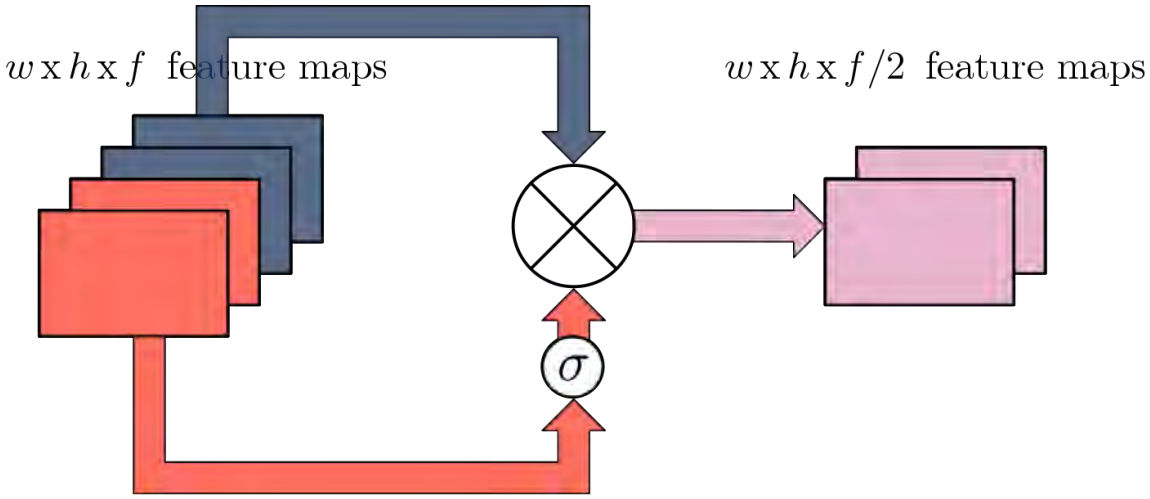

Figure 2.5.: A gated linear unit using the output of a convolutional layer as input. One half of the input feature map is used to gate the other half thereby keeping only relevant information. [5]

Dauphin et al. introduced Gated Linear Units (GLUs) [12] for CNNs to model natural language. The GLU uses one half of its input to gate the other half thus keeping only relevant information (see Figure 2.5).

More formally, the GLU splits the input into two equal sets A and B. A and B need to be computed independently. The GLU is typically used after a convolutional layer. Since each feature map is computed independently by its own filter (see Subsection 2.3.1), only the number of feature maps must be even to fulfil the independence condition. Then, B is used to gate A to get the output O:

$$O = A \otimes \sigma(B) \ , \tag{2.13}$$

where σ is the sigmoid activation function. The gates calculated from B control what information is passed on. In the context of CNNs relevant image features are passed on, whereas irrelevant image features are forgotten.

Similar to a ReLU, the GLU provides non-linear capabilities and a linear path for the gradient diminishing the vanishing gradient problem and making it applicable in deep neural networks.

[5] Figure was created by the author.

2.4. Recurrent Neural Networks (RNNs)

A disadvantage of common FFNN networks is their requirement for fixed-sized input. If the input length varies the input needs to be padded to the length of the longest input or clipped. Recurrent Neural Networks (RNNs) address this issue and are able to process variable length sequences. Hence, they are widely used in natural language processing. Moreover, RNNs recognize structures regardless of their position in the input.

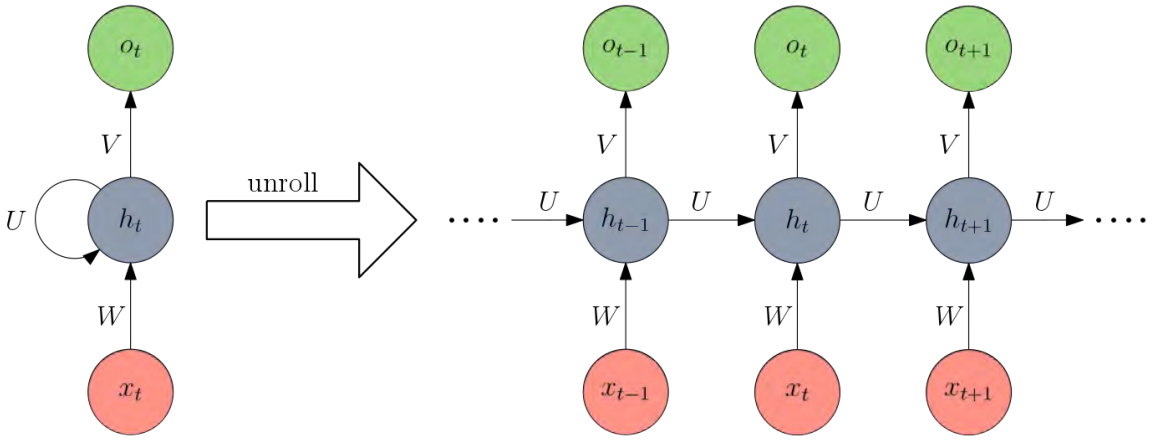

Figure 2.6.: Simple recurrent neural network and the same network unfolded over time. U, V, and W represent weight matrices. [6]

Figure 2.6 depicts a simple RNN. An RNN contains a feedback loop: the output of the previous calculation is used for the next calculation. In Figure 2.6 the hidden layer uses at each time-step the previous hidden state h_{t-1} and x_t as input:

$$h_t = f(x_t, h_{t-1}) = \sigma(Wx_t + Uh_{t-1}) \ , \tag{2.14}$$

where W is the weight matrix between the input and the hidden layer and U is the weight matrix between the hidden layer and itself. The initial hidden state h_0 can be initialised arbitrarily.

Similar to a CNN, an RNN greatly reduces the number of parameters. However, instead of sharing weights spatially, an RNN shares weights over time (see Figure 2.6).

2.4.1. Vanishing Gradient Problem

Training an RNN is similar to training a deep neural network. Since the previous output of the network influences the next output of the network, an RNN needs to be unrolled to train it (see Figure 2.6). Backpropagating through multiple layers or in this case through time either causes the gradient to vanish or to explode. This results in slow or no learning or, respectively, in divergence.

[6] Figure was created by the author.

Short-term dependencies backpropagate through only a few time-steps. Therefore, they are less affected by the vanishing gradient problem and have been shown to dominate long-term dependencies [25].

2.4.2. Long Short Term Memory (LSTM)

Long Short Term Memory (LSTM) [26] addresses the issues of the vanishing gradient problem and long term dependencies. LSTM uses a cell state which is updated or deleted independently from the output. This allows LSTMs to store long term dependencies.

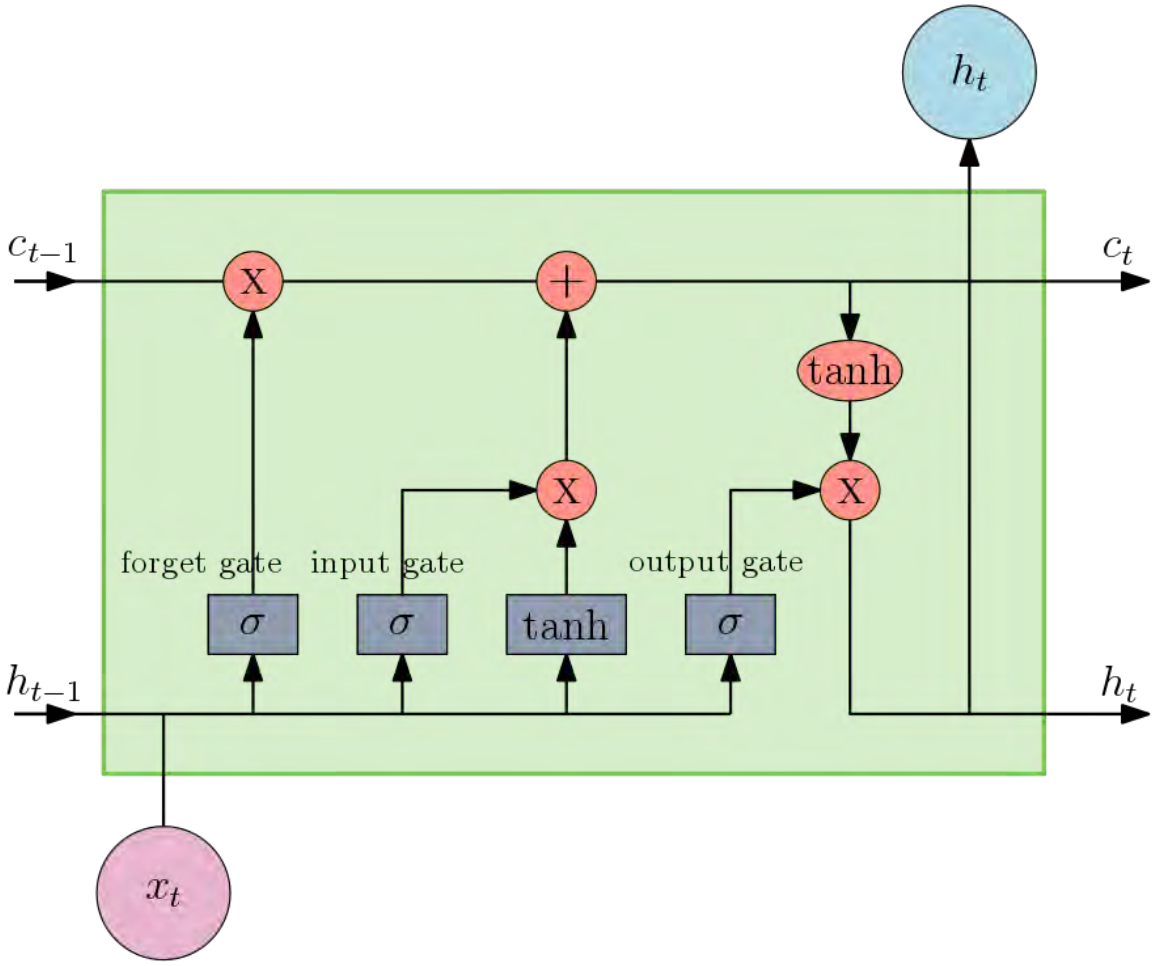

Figure 2.7.: A single LSTM cell. c_t is the cell state and h_t is the cell output. Blue boxes represent a neural network layer with its respective activation function. Red ellipses represent point-wise operations. [7]

[7] Figure was created by the author.

Figure 2.7 depicts a single LSTM cell. It receives its old cell state c_{t-1}, its old output h_{t-1}, and x_t as input. To compute the new output and cell state, a new cell state candidate is computed:

$$\hat{c}_t = tanh(W_c x_t + U_c h_{t-1}) \ . \tag{2.15}$$

A forget gate f_t specifies which old data is forgotten:

$$f_t = \sigma(W_f x_t + U_f h_{t-1}) \tag{2.16}$$

and an input gate i_t determines what new information is stored:

$$i_t = \sigma(W_i x_t + U_i h_{t-1}) \ . \tag{2.17}$$

The input gate protects the memory from noise and unnecessary information. Together, the input and the old cell state form the new cell state controlled by its respective gates:

$$c_t = f_t \odot c_{t-1} + i_t \odot \hat{c}_t \ . \tag{2.18}$$

An output gate o_t determines the importance of the input for the output:

$$o_t = \sigma(W_o x_t + U_o h_{t-1}) \ . \tag{2.19}$$

Finally, the output is computed using the new cell state (see Equation 2.18) and the output gate (see Equation 2.19): ,

$$h_t = o_t \cdot tanh(c_t) \ . \tag{2.20}$$

2.5. Generative Adversarial Networks (GANs)

Goodfellow et al. introduced Generative Adversarial Networks (GANs) [19] which have a wide range of applications such as domain transfer, synthetic data generation and refinement, super-resolution, and high-resolution image generation.

A GAN consists of two differentiable submodules: a generator G and a discriminator D that are trained interdependently. The generator models a transform function: it receives a latent noise vector z, sampled from a distribution p_z, as input. p_z is usually a normal distribution. The generator is trained to generate images $G(z)$ (or other output) resembling the training data distribution p_z.

The discriminator resembles a discriminative function or classifier: it receives the generated images $G(z)$ and the corresponding real training data x as input and is trained to differentiate between the two. The output of the discriminator $D(y)$ is the probability that D assigns to the image y that y was sampled from the true data distribution. Thus, the discriminator can be thought of as a trainable loss function for the generator. Figure 2.8 depicts this architecture.

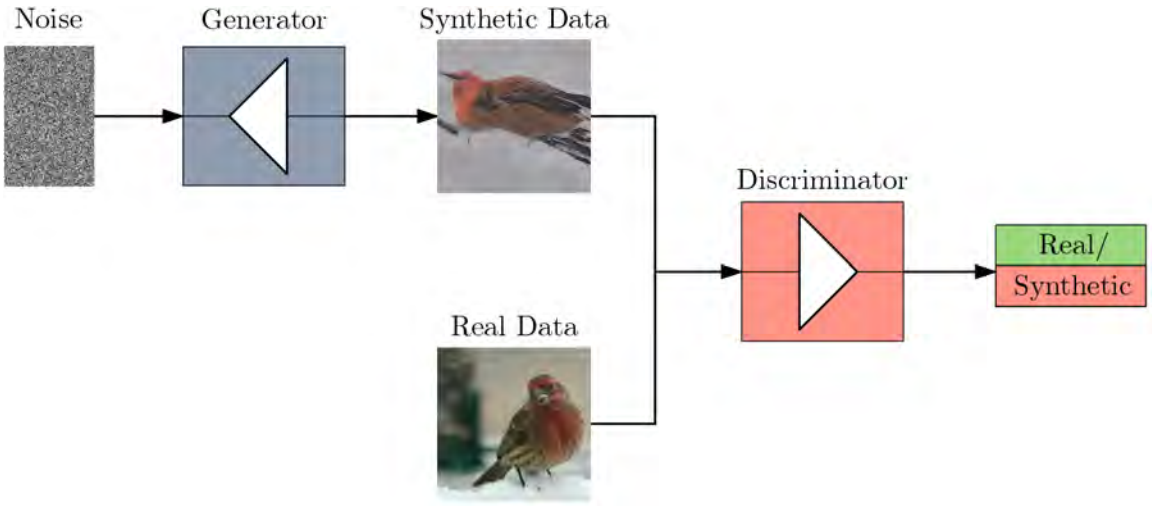

Figure 2.8.: Architecture of a simple GAN generating images from noise. [8]

The generator and discriminator compete against each other in an adversarial manner: G is trained to minimize the probability of D identifyig the images as synthetic: $D(G(z)) \approx 0$, whereas D is trained to maximize that probability: $D(G(z)) \approx 1$. This competition is expressed in a min-max function [18]:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x \sim p_R}[log(D(x))] + \mathbb{E}_{z \sim p_z}[log(1 - D(G(z)))] \ . \tag{2.21}$$

The parameters are updated using alternating stochastic gradient descent between the two models. The generator receives its error gradient from the discriminator instead of using a direct comparison of its generated samples to the training data samples, like a FFNN. Therefore, the generator never directly sees a sample of the training data.

This concept allows GANs to avoid the difficult task of comparing two probability distributions based on samples. Furthermore, because the generator only receives information about the true data distribution from gradients of the discriminator, GANs are largely unaffected by overfitting.

The GAN converges if the min-max function reaches an equilibrium. Ideally, the generator perfectly reproduces the training data distribution, i.e., $p_R = p_z$.

2.5.1. Training Instabilities

There are various difficulties when training GANs. The existence of an equilibrium is not guaranteed. Hence, convergence is not guaranteed.

[8] Figure was created by the author.

A common issue is that the discriminator often rejects samples with high confidence, especially during early phases of training where the generator is unsophisticated. Therefore, the absolute value of gradients passed on to the generator is small, hindering the generator from learning.

A severe problem is mode collapse [18]: the generator collapses to one or a few modes where multiple different inputs are mapped to the same image that the discriminator believes to be real, i.e., it has the largest discriminator output. This occurs because the gradients in the discriminator are computed independently of each other, without incorporating a similarity measure of a given mini-batch [61].

When the discriminator receives the same image or nearly identical images in a mini-batch, it quickly learns to identify this single image as fake. However, because the received images are (nearly) identical, the gradients of the discriminator are (nearly) identical as well. Thus, the generator update pushes the generator to generate a different single image instead of generating multiple images leading to an oscillatory failure state.

Various techniques have been proposed [54] [61] [1] [44] to stabilise the training procedure and generate compelling results.

2.6. Autoencoders

An autoencoder consists of an encoder and a decoder. The encoder computes an encoding of the input. It receives an input $x \in R^d$ and maps it to a latent representation $h \in R^{d'}$ using a deterministic function $f_\theta = \sigma(Wx + b)$ with parameters $\theta = \{W, b\}$. The decoder tries to reconstruct the original input from the encoding. It performs a reverse mapping using the function $y = g_{\theta'} = \sigma(W'x + b')$. Often, the parameters are constrained such that $W' = W^T$. [5]

Learning occurs in an unsupervised manner, because the label equals the input. The encoder learns an efficient data encoding which can be used for dimensionality reduction. The decoder learns to reconstruct/generate data from data encodings which can be applied in generative models.

2.6.1. Denoising Autoencoders

A further development of the autoencoder is the denoising autoencoder [72], which is more robust towards noise in the data. A denoising autoencoder tries to reconstruct noisy inputs. During training the original input x is corrupted to x' by adding v times noise on the input, for example, uncorrelated Gaussian noise for colour images. The parameter v represents the percentage of permissible corruption. Then, the autoencoder tries to reconstruct the original input x from the corrupted input x'.

3. Related Work

This chapter provides a brief overview of natural language models and generative image models. Section 3.1 introduces neural and non-neural natural language models with a focus on recent large pretrained models. Section 3.2 illustrates the three most common approaches of generative image modelling: generative adversarial networks (see Subsection 3.2.1), autoencoders (see Subsection 3.2.2), and autoregressive models (see Subsection 3.2.3), as well as further approaches (see Subsection 3.2.4).

3.1. Natural Language Models

Modelling natural language is a research area that has been around for decades and poses an important aspect of many tasks. There are non-neural approaches such as class-based *n*-gramm models with essentially a hidden Markov model (HMM) [7] or a spectral algorithm [65], structural correspondence learning [6], etc.

Neural approaches, such as deep RNNs, have been around for a couple of years. Recently, with ELMo [49], ULMFiT [27], and with the OpenAI GPT [55] and BERT [15] using the transformer architecture [70], new large pretrained language representation models were developed, each advancing the state of the art for multiple NLP tasks.

The idea is to pretrain those models unsupervised on a large amount of data, at least 100 million words+. BERT, for example, was trained on the BooksCorpus (800 million words) [87] and English Wikipedia (2500 words). Then, the models are fine-tuned on the respective downstream task, only needing to learn a fraction of their parameters.

3.2. Generative Image Models

While there has been substantial work for years in the field of image-to-text translation, such as image caption generation [3] [17] [79], only recently the inverse problem came into focus: text-to-image generation. Generative image models require a deep understanding of spatial, visual, and semantic world knowledge and have many applications for artists or graphic designers, such as photo-editing, computer-aided design, etc. Conditional image synthesis conditions generation on additional input.

Generative image models can be divided into three categories: Generative Adversarial Networks (GANs) [19] jointly train a generator for synthesizing images and a discriminator for classifying them as real or fake; Variational Autoencoders (VAEs) [34] use probabilistic

graphical models with the goal to maximize the lower bound of data-likelihood; and Autoregressive Models [47] which factorise the joint distribution of images into per-pixel factors and condition each pixel on all previous pixels.

An alternate approach of discriminating generative image models is by their type of input: for example, image-to-image translation with pixel-wise semantic scene layouts as input can be done with a conditional GAN [30], an encoder-decoder [59], or with a CNN modelled as a cascaded refinement network [9]. Apart from various types of image inputs [11], there are plain text [57] [84], processed text, such as scene graphs [32], and text with additional information, for example, segmentation masks and bounding boxes for characters and objects [22] or object location constraints [58], inputs.

3.2.1. Generative Adversarial Networks (GANs)

Reed et al. [57] use a GAN with a direct text-to-image approach and have shown to generate images highly related to the text's meaning. Reed et al. [56] further developed this approach by conditioning the GAN additionally on object locations. Zhang et al. built on Reed et al.'s direct approach developing Stacked Generative Adversarial Networks (StackGAN) [84] generating 256x256 photo-realistic images from detailed text descriptions. Although StackGAN yields remarkable results on specific domains, such as birds or flowers, it struggles when many objects and relationships are involved.

Zhang et al. [85] improved StackGAN by arranging multiple generators and discriminators in a tree-like structure, allowing for more stable training behaviour by jointly approximating multiple distributions. Xu et al. [81] introduced a novel loss function and fine-grained word attention into the model.

Recently, a number of works built on Xu et al.'s [81] approach: Cheng et al. [10] employed spectral normalisation [44] and added global self-attention to the first generator; Qiao et al. [53] introduced a semantic text regeneration and alignment module thereby learning text-to-image generation by redescription; Li et al. [38] added channel-wise attention to Xu et al.'s spatial word attention to generate shape-invariant images when changing text descriptions; Cai et al. [8] enhanced local details and global structures by attending to related features from relevant words and different visual regions; Yin et al. [82] focused on disentangling the semantic-related concepts and introduced a contrasive loss to strengthen the image-text correlation; and Zhu et al. [86] refined Xu et al.'s fine-grained word attention by dynamically selecting important words based on the content of an initial image.

Instead of using multiple stages or multiple GANs, Li et al. [39] used one generator and three independent discriminators to generate multi-scale images conditioned on text in an adversarial manner. Johnson et al. [32] introduced a GAN that receives a scene graph consisting of objects and their relationships as input and generates complex images with many recognizable objects. However, the images are not photo-realistic. Qiao et

al. [52] introduced LeicaGAN which adopts text-visual co-embeddings to convey the visual information needed for image generation.

3.2.2. Autoencoders

Autoencoders not only generate images but interpret them. However, autoencoders tend to generate blurry images when being used with popular error functions, such as MAE or MSE. Thus, novel approaches use different error functions. Furthermore, estimating likelihoods in a high-dimensional image space is very difficult [68].

Snell et al. [64] use deterministic and stochastic autoencoders with multiscale structural-similarity score (MS-SSIM) [76], a loss function calibrated to human perceptual judgments of image quality. Dorta et al. [16] use a hierarchy of VAEs analogous to a Laplacian pyramid. Each VAE models a single pyramid level and is conditioned on the coarser levels. Additionally, a novel loss function is used, allowing the latent Gaussian to have an arbitrary mean and variance while still being efficiently trainable and samplable. The Laplacian framework can also be applied to GANs to generate natural images in a coarse to fine fashion, see [14].

3.2.3. Autoregressive Models

Oord et al. developed PixelCNN [46] an autoregressive model to generate images conditioned on any vector, for example, descriptive labels, tags, or latent embeddings created by other networks. To do so PixelCNN uses autoregressive connections to model images pixel by pixel, decomposing the joint image distribution as a product of conditionals. This approach requires one network evaluation per pixel. Hence, inference is costly: O(N) for N pixels. Reed et al. [58] improve on this approach by modelling certain pixel groups as conditionally independent via object locations. They achieve O(log N) sampling instead of O(N) sampling, thereby enabling the practical generation of 512x512 images.

Gupta et al. introduced CRAFT (Composition, Retrieval and Fusion Network) [22] a network capable of generating short cartoon videos from novel captions. CRAFT learns from densely annotated video clips. However, CRAFT requires segmentation masks and bounding boxes for characters and objects as well a clean background. Moreover, CRAFT does not generate frames pixel-by-pixel. Instead, it sequentially composes a scene layout and retrieves entities from a video database.

3.2.4. Further Approaches

Further approaches encompass generative image modelling using an RNN with spatial LSTM neurons [67]; multiple layers of convolution and deconvolution operators trained with Stochastic Gradient Variational Bayes [36]; an encoder de-rendering an image to an XML scene description with a deterministic rendering function as decoder [77]; a probabilistic programming language for scene understanding with fast general-purpose inference machinery [35]; and generative ConvNets [78].

4. Method

This chapter presents reasoning behind our choice of base architecture, several attention models, approaches of interpreting and combining attention, and spectral normalisation. Section 4.1 discusses the choice of the AttnGAN [81] as our base architecture. Section 4.2 presents the structure of the AttnGAN and the novel loss function it introduces. In Section 4.3 we present global (Subsection 4.3.1), local (Subsection 4.3.2), and light-weight (Subsection 4.3.3) self-attention for CNNs.

Section 4.4 repurposes linear attention (Subsection 4.4.1) and grid attention (Subsection 4.4.2) to be used as sentence attention. Section 4.5 focuses on incorporating the attention models of Section 4.3 and Section 4.4 in the AttnGAN. This includes the upsampling block (Subsection 4.5.1) of the network with no previous attention model and extending the attention module (Subsection 4.5.2) to mix the pre-existing word attention with other attention models.

4.1. Motivation

This section provides insights of the choice of our base architecture, text-encoder, and attention models. First, we decided on the general direction of our approach. Most pre-existing work in the context of generative image modelling falls into three categories: Autoencoder, Autoregressive Model, or Generative Adversarial Network (GAN).

Autoencoders typically require the same input and output format. With text-to-image generation the input and the output are fundamentally different. Training an autoencoder with a text representation as input and an image representation as output will not work. Using text and image as input may lead to a multimodal representation in the autoencoder. However, this would still require to bridge the gap between the text representation and the learned multimodal representation.

Autoregressive models typically model images pixel by pixel which makes them impractical for larger images. Advanced modelling techniques usually require additional input, like object locations, segmentation masks, bounding boxes, etc. This work focuses on large images (256x256 pixels). Thus, we decided against autoregressive models.

While GANs have training stability issues, they excel at generating data. Furthermore, by using a discriminator network as a trainable loss function, they are a able to avoid having to use complicated loss functions for images. Therefore, and because of their success in this domain we chose GANs.

A number of existing GAN approaches [86] [82] [52] [8] [38] [10] [53] [81] employ the concept of stacked GANs [84] [85]. Following the approach of recent models [86] [82] [8] [38] [53] [10] we chose Xu et al.'s AttnGAN [81] as our base architecture, ensuring better comparability through a common baseline. Furthermore, the network allows us to investigate in the introduction of different attention models. With the recent success of self-attention in the field of generative image modelling, we mainly focus on self-attention.

We did some minor experiments with large pretrained text-encoders, namely BERT [15]. While they had major success in a number of other NLP-tasks, we found them impractical in our context. In the other tasks the text representation is almost the solution which is then obtained by adding a softmax layer or a small network.

While a good text representation is necessary for text-to-image generation, the more challenging issue is the translation of the text representation to the image representation. Due to their versatility, these pretrained models are large, for example, BERT with 100 million parameters in its smaller version. Therefore, spending a lot of resources on them while retaining a major part of the problem is impractical.

4.2. AttnGAN

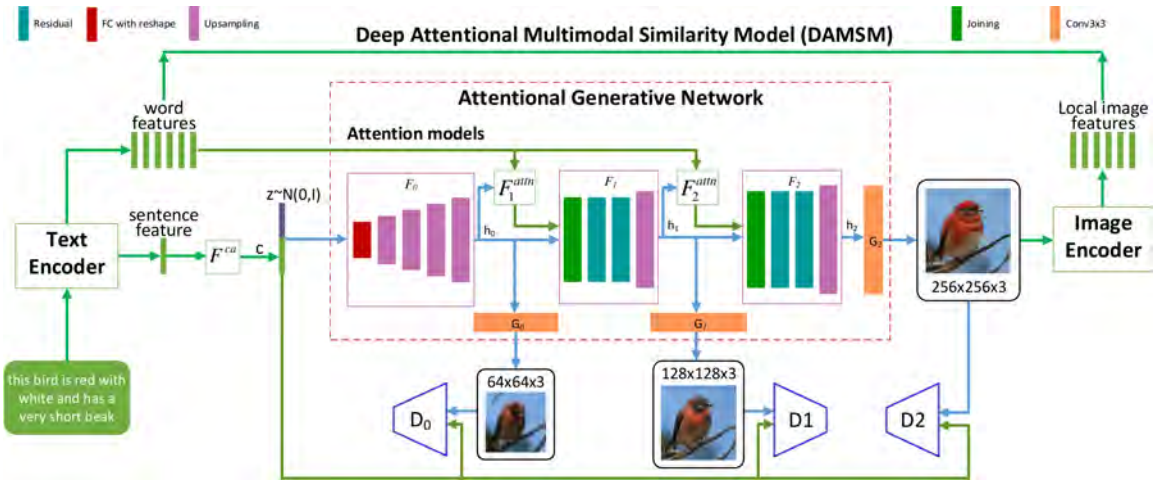

Figure 4.1.: The AttnGAN architecture. Each attention model automatically retrieves the conditions (i.e., the most relevant word vectors) for generating different subregions of the image. The DAMSM provides a fine-grained image-text matching loss for the generative networks. [1]

[1] Figure taken from [81].

The AttnGAN [81] consists of two models: an attentional generative network consisting of stacked GANs generating images in a small-to-large scale fashion (see Subsection 4.2.1) and a Deep Attentional Multimodal Similarity Model (DAMSM, see Subsection 4.2.2) computing a fine-grained image-text matching loss. Figure 4.1 depicts the architecture of the AttnGAN. Figure 4.2 and Figure 4.3 visualize submodules.

4.2.1. Attentional Generative Model

The attentional generative model consists of m generators $(G_0, G_1, ..., G_{m-1})$, which receive the image feature vectors $(h_0, h_1, ..., h_{m-1})$ as input and generate images of small-to-large scales $(\hat{y_0}, \hat{y_1}, ..., \hat{y_{m-1}})$.

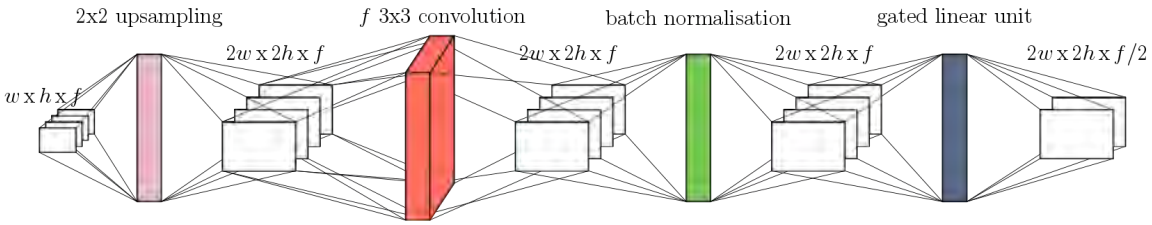

Figure 4.2.: The architecture of an upsampling block in Figure 4.1. [2]

First, a deep bidirectional LSTM encoder encodes the input sentence into a global sentence vector s and a word matrix w. Conditioning Augmentation F^{CA} [84] converts the sentence vector into the conditioning vector.

Conditioning augmentation [84] is a technique introduced by Zhang et al. which generates additional conditioning variables. Instead of using a non-linear transformation on the sentence vector, conditioning augmentation uses s to sample from an independent Gaussian distribution:

$$\mathcal{N}\big(\mu(s), \sum(s)\big) \ , \tag{4.1}$$

where the mean and the diagonal covariance matrix are functions of s. This yields more training pairs, thereby mitigating the problem of discontinuities in the latent data manifold due to a high-dimensional latent space for text-embeddings (usually > 100) and sparse training pairs. Thus, it encourages robustness to small perturbations along the conditioning manifold.

One textual description usually corresponds to a number of images. To facilitate that variety, a regularisation term is added to the objective function of the generator during training. The regularisation term is the Kullback-Leiber (KL) divergence between the conditioning Gaussian and the standard Gaussian distribution:

$$D_{\mathrm{KL}}\big(\mathcal{N}\big(\mu(s), \sum(s)\big) || \mathcal{N}(0, 1)\big) \ . \tag{4.2}$$

[2] Figure was created by author.

A first network F_0, consisting of a fully connected layer and then upsampling blocks (see Figure 4.2, Figure 4.1), receives the conditioning vector $F^{CA}(s)$ and noise z, sampled from a standard normal distribution, as input and computes the first image feature vector h_0:

$$h_0 = F_0\big(z, F^{CA}(s)\big) \ . \tag{4.3}$$

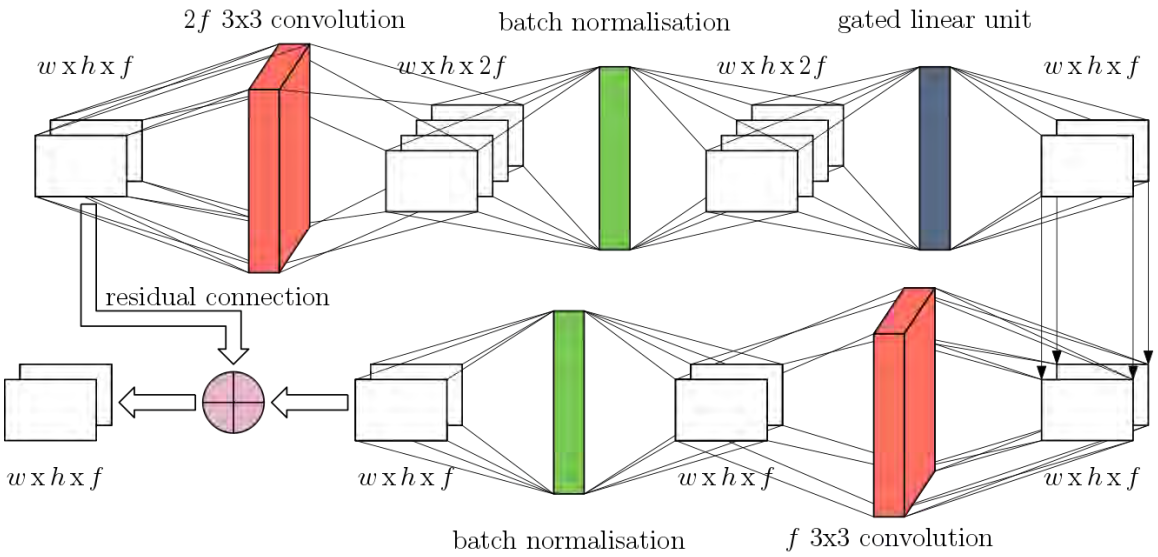

Figure 4.3.: The architecture of a residual block in Figure 4.1. [3]

Each generator G_i is a simple 3x3 convolutional layer that receives the image feature vector h_i as input to compute the image $\hat{y}_i$:

$$\hat{y}_i = G_i(h_i) \ . \tag{4.4}$$

The image feature vectors h_i, except for h_0, are computed by the network F_i consisting of multiple residual blocks and an upsampling block (see Figure 4.3, Figure 4.2, Figure 4.1). F_i receives the previous image feature vector h_{i-1} and the result of the i^{th} attentional model F_i^{attn} as input:

$$h_i = F_i\big(h_{i-1}, F_i^{attn}(w, h_{i-1})\big) \ . \tag{4.5}$$

The attentional model F_i^{attn} receives the previous image feature vector h_{i-1} and the word matrix w of the text-encoder as input and computes an attention map. First, the word vectors are converted into a common semantic space using a perceptron layer U:

$$w' = Uw \ . \tag{4.6}$$

For each subregion k of the image a word-context vector $c_{i,k}$ is computed. The word-context vector is a dynamic representation of word vectors that are relevant to the subregion of the

[3] Figure was created by author.

image. Each column k in the image feature vector h_i is a feature vector of the subregion k of the image. The word-context vector $c_{i,k}$ is computed by:

$$c_{i,k} = \sum_{l=0}^{T-1} beta_{i,k,l} w'_i, \quad \text{where} \quad \beta_{i,k,l} = \frac{exp(s_{i,k,l})}{\sum_{z=0}^{T-1} exp(s_{i,k,z})} \quad \text{and} \quad s_{i,k,l} = h_{i,j}^T w'_k \ . \tag{4.7}$$

$\beta_{i,k,l}$ indicates the weight the i^{th} model attends to the l^{th} word when generating subregion k. Then, the next image feature vector is computed according to Equation 4.5. The final image $\hat{y_{m-1}}$ is generated by the last generator G_{m-1} from the final image feature vector h_{m-1} (Equation 4.4).

To generate realistic images with multiple levels (i.e., sentence level and word level) of conditions, the final objective function of the attentional generative network is defined as:

$$L = L_G + \lambda L_{\text{DAMSM}}, \quad where \ L_G = \sum_{i=0}^{m-1} L_{G_i} \ . \tag{4.8}$$

Here, λ is a hyperparameter to balance the two terms. We follow the authors recommendation of a $\lambda = 5$ for the CUB dataset [74]. The first term is the GAN loss that jointly approximates conditional and unconditional distributions. At the i^{th} stage of the AttnGAN, the generator G_i has a corresponding discriminator D_i. The adversarial loss for G_i is defined as:

$$L_{G_i} = \underbrace{-\frac{1}{2}\mathbb{E}_{\hat{y}_i \sim P_{G_i}}\left[log(D_i(\hat{y}_i))\right]}_{\text{unconditional loss}} \underbrace{-\frac{1}{2}\mathbb{E}_{\hat{y}_i \sim P_{G_i}}\left[log(D_i(\hat{y}_i, s))\right]}_{\text{conditional loss}}, \tag{4.9}$$

where the unconditional loss determines whether the image is real or fake while the conditional loss determines whether the image and the sentence match or not. Alternately to the training of G_i, each discriminator D_i is trained to classify the input into the class of real or fake by minimizing the cross-entropy loss defined by:

$$\begin{aligned} L_{D_i} = & \underbrace{-\frac{1}{2}\mathbb{E}_{y_i \sim P_{data_i}}\left[log(D_i(y_i))\right] - \frac{1}{2}\mathbb{E}_{\hat{y}_i \sim P_{G_i}}\left[log(1 - D_i(\hat{y}_i))\right]}_{\text{unconditional loss}} + \\ & \underbrace{-\frac{1}{2}\mathbb{E}_{y_i \sim P_{data_i}}\left[log(D_i(y_i, s))\right] - \frac{1}{2}\mathbb{E}_{\hat{y}_i \sim P_{G_i}}\left[log(1 - D_i(\hat{y}_i, s))\right]}_{\text{conditional loss}}, \end{aligned} \tag{4.10}$$

where y_i is from the true image distribution P_{data_i} at the i^{th} scale, and $\hat{y}_i$ is from the model distribution P_{g_i} at the same scale. Discriminators of the AttnGAN are structurally disjoint, so they can be trained in parallel. Each of them focuses on a single image scale. The second term of Equation 4.8, L_{DAMSM}, is a fine-grained word-level image-text matching loss computed by the DAMSM, elaborated in Subsection 4.2.2. [81]

4.2.2. Deep Attentional Multimodal Similarity Model (DAMSM)

This subsection outlines the Deep Attentional Multimodal Similarity Model (DAMSM), for details regarding the corresponging loss function we refer to [81]. The DAMSM learns two neural networks that map subregions of the image and words of the sentence to a common semantic space, thus measuring the image-text similarity at the word level to compute a fine-grained loss for image generation.

The text-encoder is a bidirectional LSTM encoder. The image encoder is built upon a pretrained Inception-v3 model [66]. The weights of the Inception-v3 model remain fixed. Added perceptron layers extract visual feature vectors for each subregion of the image and a global image vector.

Similar to Equation 4.7 of Subsection 4.2.1, with an extra hyperparameter determining how attention is paid to features of relevant subregions, the word-context vectors c for each subregion are computed. Then, the relevance between the i^{th} word and the image is computed:

$$R(c_i, w_i) = (c_i^T w_i)/(\|c_i\|\|w_i\|) \ . \tag{4.11}$$

Finally, the attention-driven image-text matching score between the entire image (Q) and the whole text description (D) is defined as:

$$R(Q, D) = log\Big(\sum_{i=1}^{T-1} exp(\gamma_2 R(c_i, w_i))\Big)^{1/\gamma_2} \ . \tag{4.12}$$

This attention-driven image-text matching score is used to define the loss function L_{DAMSM}, for details we refer to [81].

4.3. Self-Attention Models

4.3.1. Global Self-Attention

Recently, Zhang et al. [83] introduced a self-attention model. It addresses the issue of spatial locality in CNNs by modelling long-range dependencies for image generation.

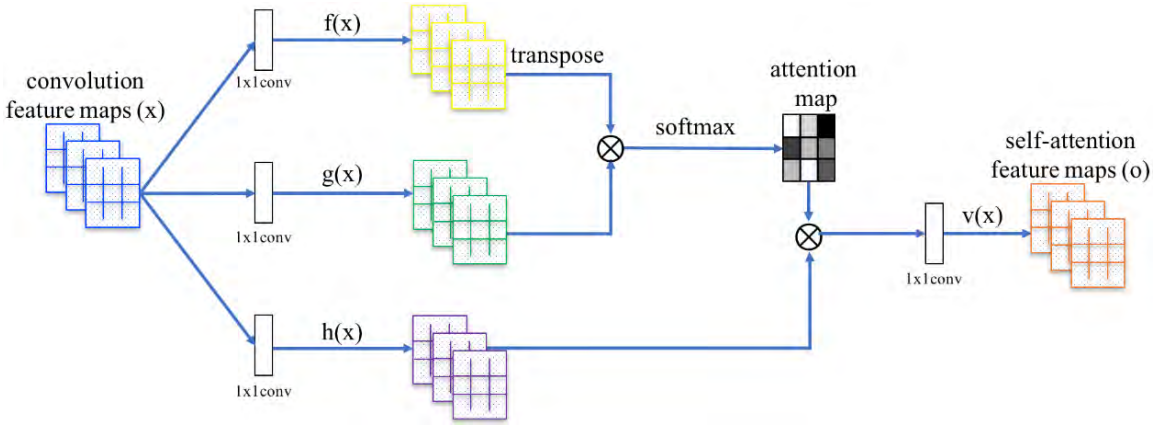

Figure 4.4.: Global self-attention module. [4]

Figure 4.4 depicts the self-attention model. The feature maps of a convolutional layer of size $w \times h \times c$ are fed to three individual 1x1 convolutional layers yielding a query of size $\frac{w}{k} \times \frac{h}{k} \times c$, a key of size $\frac{w}{k} \times \frac{h}{k} \times c$, and a value of size $w \times h \times c$. While the value maintains the number of channels, the query and the key usually reduce the number of channels by a factor of k to save memory consumption in the following step. k has to be a power of 2 and is commonly around 8.

The query and the key are both reduced along the spatial dimension rendering matrices of size $\frac{w}{k} \cdot \frac{h}{k} \times c = \frac{w \cdot h}{k^2} \times c$. Then, the query is transposed and multiplied to the key yielding a matrix of size $\frac{w \cdot h}{k^2} \times \frac{w \cdot h}{k^2}$. A softmax generates an attention-map.

Lastly, the attention map is multiplied by the value and then convolved by a 1x1 convolutional layer resulting in a global self-attention map of the original input size ($w \times h \times c$).

The authors recommend to scale this final self-attention map by a learnable parameter γ and add it back to the input. Thus, we can interpret this attention map as a scaled heightmap. γ is initialised with zero allowing the network to first rely on local cues in the neighbourhood and then to gradually learn to assign more importance to non-local evidence, i.e., the global self-attention.

[4] Figure taken from [83].

An issue of global self-attention is its memory consumption. The factor of k mitigates that problem to some extent, because it influences the matrix size quadratically. However, both the width and height of the feature map have a quadratic influence. Therefore, adjusting the spatial size, in both width and height, has an impact of a power of 4. Hence, global self-attention becomes unviable for large feature maps.

For example: given an input of size 128x128x16 the intermediate matrix with a k of 1 has the size: $(128 \cdot 128 \cdot 16)^2 \cdot 32\,\text{bit} = 256\,\text{GiB}$. With a recommended k of 8, the size shrinks by $k^2 = 256$ down to 1 GiB. However, that size still has to be multiplied by the batch size. Thus, we downsample inputs of a width and/or height of 128 or higher to a size of 64 using mean-pooling. The resulting self-attention map is upsampled using the nearest strategy.

4.3.2. Local Self-Attention

Ramachandran et al. [48] proposed a self-attention model similar to the model in Subsection 4.3.1. However, while the former is a global attention model making it unviable for large spatial inputs, Ramachandran et al. focus on a local model. Similar to a convolution, the proposed attention mechanism extracts a local region of pixels $ab \in \mathcal{N}_k(i,j)$ for each pixel x_{ij} and a given spatial extent k. An output pixel y_{ij} computes as follows:

$$y_{ij} = \sum_{a,b \in \mathcal{N}_k(i,j)} \text{softmax}_{ab}(q_{ij}^T k_{ab}) v_{ab} \ . \tag{4.13}$$

$q_{ij} = W_Q x_{ij}$ denotes the queries, $k_{ab} = W_K x_{ab}$ the keys, and $v_{ab} = W_V x_{ab}$ the values, each obtained via linear transformations W of the pixel ij and their neighbourhood pixels. Figure 4.5 depicts this mechanism.

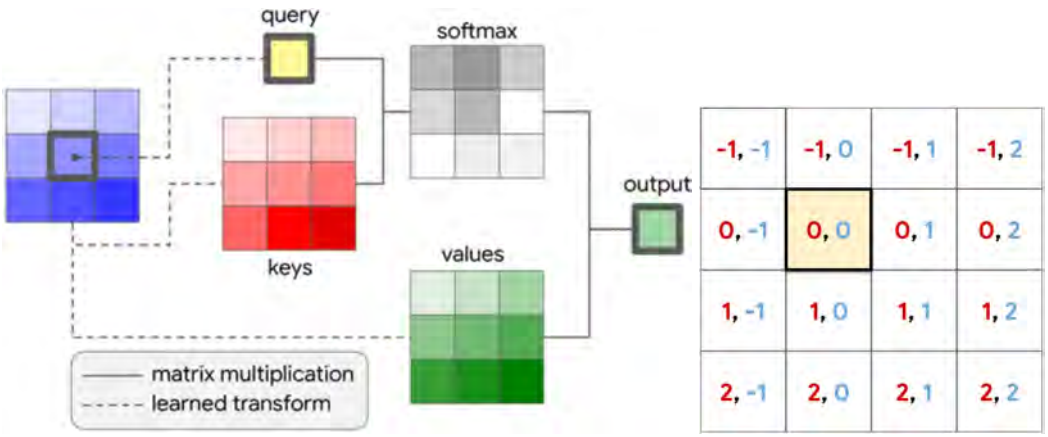

Figure 4.5.: Left: The self-attention module with a kernel size of 3. Right: Relative distances. The format of the distances is *row offset*, *column offset*. [5]

[5] Figure taken from [48].

The advantage over a simple convolution is that each pixel value is aggregated with a convex convolution of value vectors with mixing weights (softmax_{ab}) parametrised by content interactions. Furthermore, the mechanism can easily be used as a multi-headed attention mechanism by partitioning the pixel-features x_{ij} depth-wise into N groups. Then, each group separately computes single-headed attention with different transformations W_Q, W_K, and W_V. Concatenating the output of each head yields the final output y_{ij}. This allows to learn multiple distinct representations of the input.

The authors intended to replace all convolutional layers, excluding 1x1 convolutions, in a network with their attention mechanism. But, in its current form the attention encodes no positional information. Therefore, permutations are equivariant limiting its expressivity for vision tasks. The authors propose a second mechanism addressing this issue with relative attention, i.e., attention with 2D relative positional embeddings.

Relative attention uses a relative distance of ij to each position $ab \in \mathbb{N}_k(i,j)$. The relative distance is factorised across dimensions, i.e., each position ab is assigned two values: a row offset $a-i$ and a column offset $b-j$. Figure 4.5 depicts an example. The row- and column offsets refer to embeddings r_{a-i} and r_{b-j}, respectively. Each embedding is half the output dimension. $r_{a-i,b-j}$ refers to the concatenation of both embeddings. Then, the relative attention is defined as:

$$y_{ij} = \sum_{a,b \in \mathcal{N}_k(i,j)} \text{softmax}_{ab}(q_{ij}^T k_{ab} + q_{ij}^T r_{a-i,b-j}) v_{ab} \ . \tag{4.14}$$

Thus, in addition to its content, each element $a,b \in \mathcal{N}_k(i,j)$ is also modulated by its relative distance. Therefore, this mechanism is translation equivariant like convolutions. Furthermore, the parameter count is independent of the size of the spatial extent, whereas convolutional parameters grow quadratically. The computational cost of attention grows slower with spatial extent compared to convolutions, for example, if $d_{in} = d_{out} = 128$ a convolutional layer with a kernel size of 3 has the same computational cost as an attention layer with a kernel size of 19.

4.3.3. Squeeze-and-Excitation Blocks

Recently, Hu et al. introduced Squeeze-and-Excitation (SE) [28] blocks. Instead of focusing on the spatial component of CNNs, SE blocks aim to improve the channel component by explicitly modelling interdependencies among channels via channel-wise weighting. Thus, they can be interpreted as a light-weight self-attention function on channels.

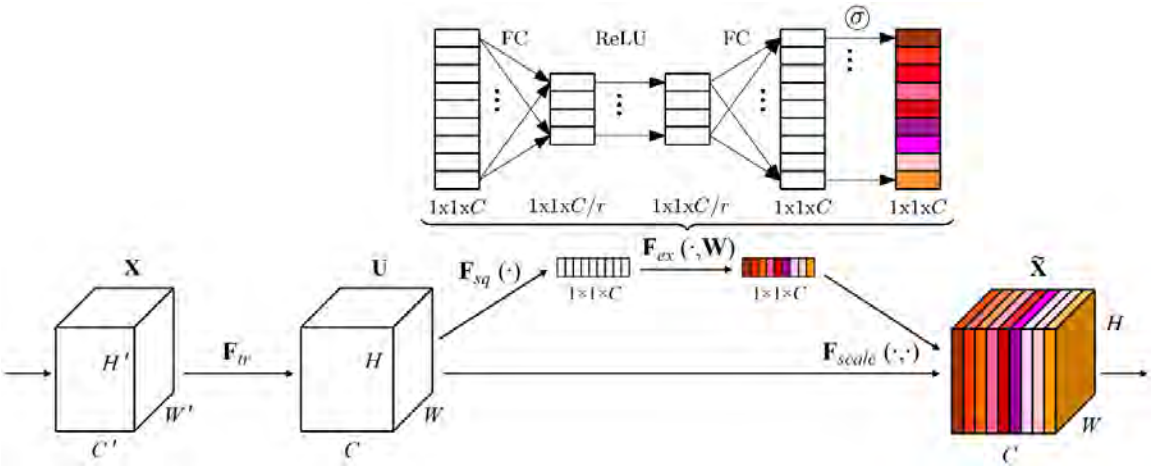

Figure 4.6.: A convolution followed by a squeeze-and-excitation block. [6]

Figure 4.6 depicts the structure of an SE block. First, a transformation F_{tr}, which is typically a convolution, outputs the feature map U. Because convolutions use local receptive fields, each entry of U is unaware of contextual information outside its region. A corresponding SE-block addresses this issue by performing a feature recalibration.

A squeeze operation aggregates the feature maps of U across the spatial dimension (H x W) yielding a channel descriptor. The proposed squeeze operation is mean-pooling across the entire spatial dimension of each channel. The resulting channel descriptor serves as an embedding of the global distribution of channel-wise features.

A following excitation operation F_{ex} aims to capture channel-wise dependencies, specifically non-linear interaction among channels and non-mutually exclusive relationships. The latter allows multiple channels to be emphasized. The excitation operation is a simple self-gating operation with a sigmoid activation function:

$$F_{ex}(z, W) = \sigma(g(z, W)) = \sigma(W_2\delta(W_1 z)) \ , \tag{4.15}$$

where δ refers to the ReLU activation function, $W_1 \in \mathbb{R}^{\frac{C}{r}xC}$, and $W_2 \in \mathbb{R}^{Cx\frac{C}{r}}$. To limit model complexity and increase generalisation, a bottleneck is formed around the gating mechanism: a Fully Connected (FC) layer reduces the dimensionality by a factor of r. A second FC layer restores the dimensionality after the gating operation. The authors recommend an r of 16 for a good balance between accuracy and complexity ($\sim$ 10% parameter increase on ResNet-50). Ideally, r should be tuned for the intended architecture.

The excitation operation F_{ex} computes per-channel modulation weights. These are applied to the feature maps U performing an adaptive recalibration.

[6] Figure adapted from [28].

4.4. Sentence Attention

4.4.1. Linear Attention

Jetley et al. [31] used a simple additive linear attention mechanism for their image classification model. We use this mechanism to obtain a single feature map highlighting important regions of the currently generated intermediate image based on the sentence embedding.

The mechanism assumes a compatibility function $C(\hat{L}^s, g)$ for linear transformed feature maps $\hat{L}^s$ and a global feature vector g. The linear transformation maps feature maps L^s to the dimensionality of g, i.e., it matches the number of channels. The authors suggest either the repurposed alignment model:

$$c_i^s = \langle u, \hat{l}_i^s + g \rangle \ , \tag{4.16}$$

where u is a weight vector that learns the universal set of relevant features, or the dot product:

$$c_i^s = \langle \hat{l}_i^s + g \rangle \tag{4.17}$$

for the compatibility function. Given a compatibility function $C(\hat{L}^s, g)$, compatibility scores are computed and then normalized using the softmax function:

$$a_i^s = \frac{\exp(c_i^s)}{\sum_j^n \exp c_j^s} \ . \tag{4.18}$$

The final output is obtained by element-wise averaging over the normalised compatibility scores:

$$a_i^s = \sum_i^n a_i^s \cdot l_i^s \ . \tag{4.19}$$

4.4.2. Grid Attention

Schlemper et al. [63] introduced a grid attention block using Attention Gates (AGs). The block can easily be included in any CNN. The proposed AG mechanism identifies salient image regions and prunes feature responses to preserve only relevant activations. Identical to Subsection 4.4.1, the block was originally designed for image classification. We use it to obtain a single feature map highlighting important regions of the currently generated intermediate image based on the sentence embedding.

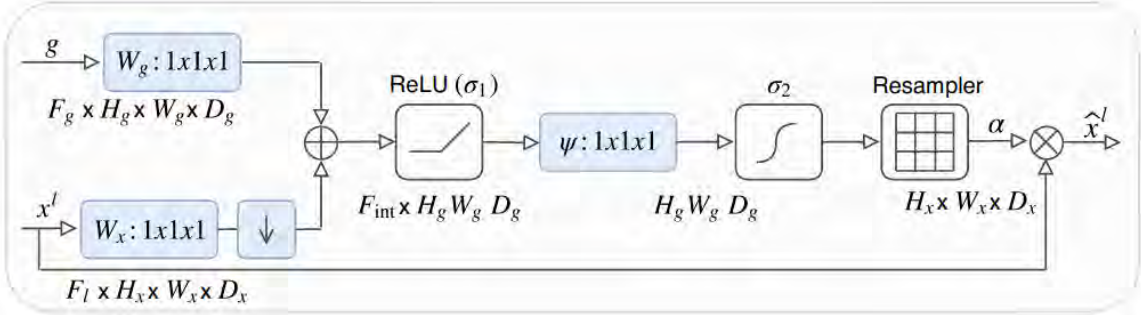

Figure 4.7.: Model of the grid attention block using Attention Gates. [7]

Figure 4.7 illustrates the grid attention block. It expects two inputs: the gating signal g and feature maps x^l. Both inputs are considered jointly to attend features that are most relevant. The number of channels of the gating signal and the feature maps must match. The additive gating mechanism is:

$$q^l_{att,i} = \psi^T(\sigma_1(W_x^T x_i^l + W_g^T g + b_{xg})) + b_\psi \tag{4.20}$$

$$\alpha^l = \sigma_2(q^l_{att}(x^l, g; \theta_{att})) \ . \tag{4.21}$$

$\sigma_1(x)$ is the ReLU and $\sigma_2(x)$ is a normalisation function. The normalisation function can be a sigmoid, softmax, or any activation function with the output range $[0, 1]$. θ_{att} denotes a set of parameters: linear transformations W_x, W_g, φ and bias terms b_φ, b_{xg}. The output of the AG mechanism α^l represents coefficients between zero and one and determines salient image regions and prunes feature responses. The final output of the grid attention block is: $\alpha^l \cdot x^l$.

4.5. Models

This section illustrates how the different attention models presented in Section 4.3 and Section 4.4 are incorporated into the base model presented in Section 4.2. We view the parts of the network independently. Subsection 4.5.1 covers the upsampling block, which is primarily used in the first part of the network, and Subsection 4.5.2 illustrates the attention module used by each generator except the first.

4.5.1. Upsampling Block

The original upsampling block is described in Subsection 4.2.1 and illustrated in Figure 4.2. We incorporate global (see Subsection 4.3.1) or local (see Subsection 4.3.2) self-attention by either adding the self-attention layer before the convolutional layer or by replacing the convolutional layer with the self-attention layer.

[7] Figure taken from [63].

When replacing the convolutional layer with a self-attention layer, either local or global, we do not add scaling or residual connections (see Figure 4.8 top). When adding local self-attention, we simply add the new block (see Figure 4.8 mid). When adding the global self-attention layer, we follow the authors [83] recommendation of scaling the output of the self-attention layer and adding its input back to it, i.e., creating a residual connection (see Figure 4.8 bottom).

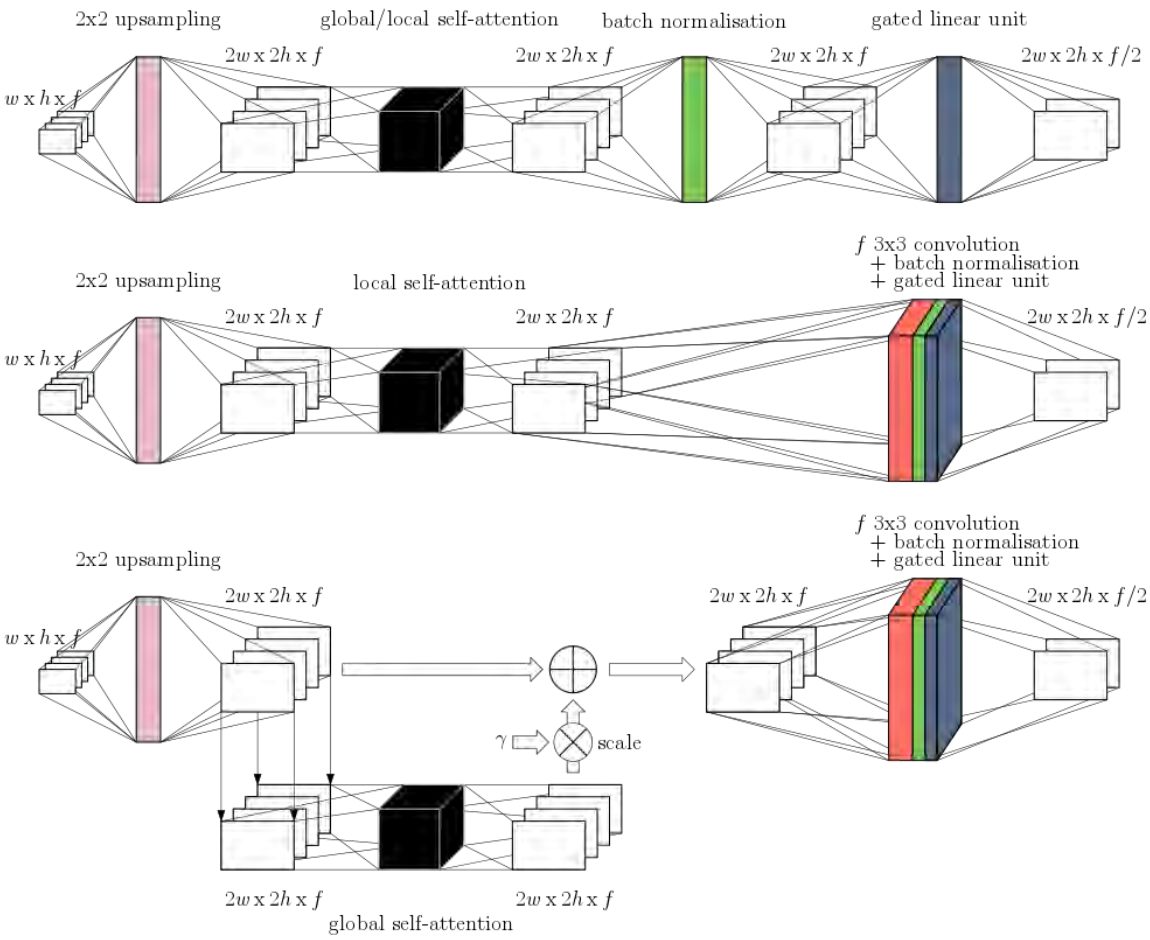

Figure 4.8.: Top: upsampling block with global/local self-attention instead of convolutions.
Middle: upsampling block with an added local self-attention block.
Bottom: upsampling block with scaled global self-attention that is added back to the input. [8]

[8] Figure was created by author.

4.5.2. Attention Module

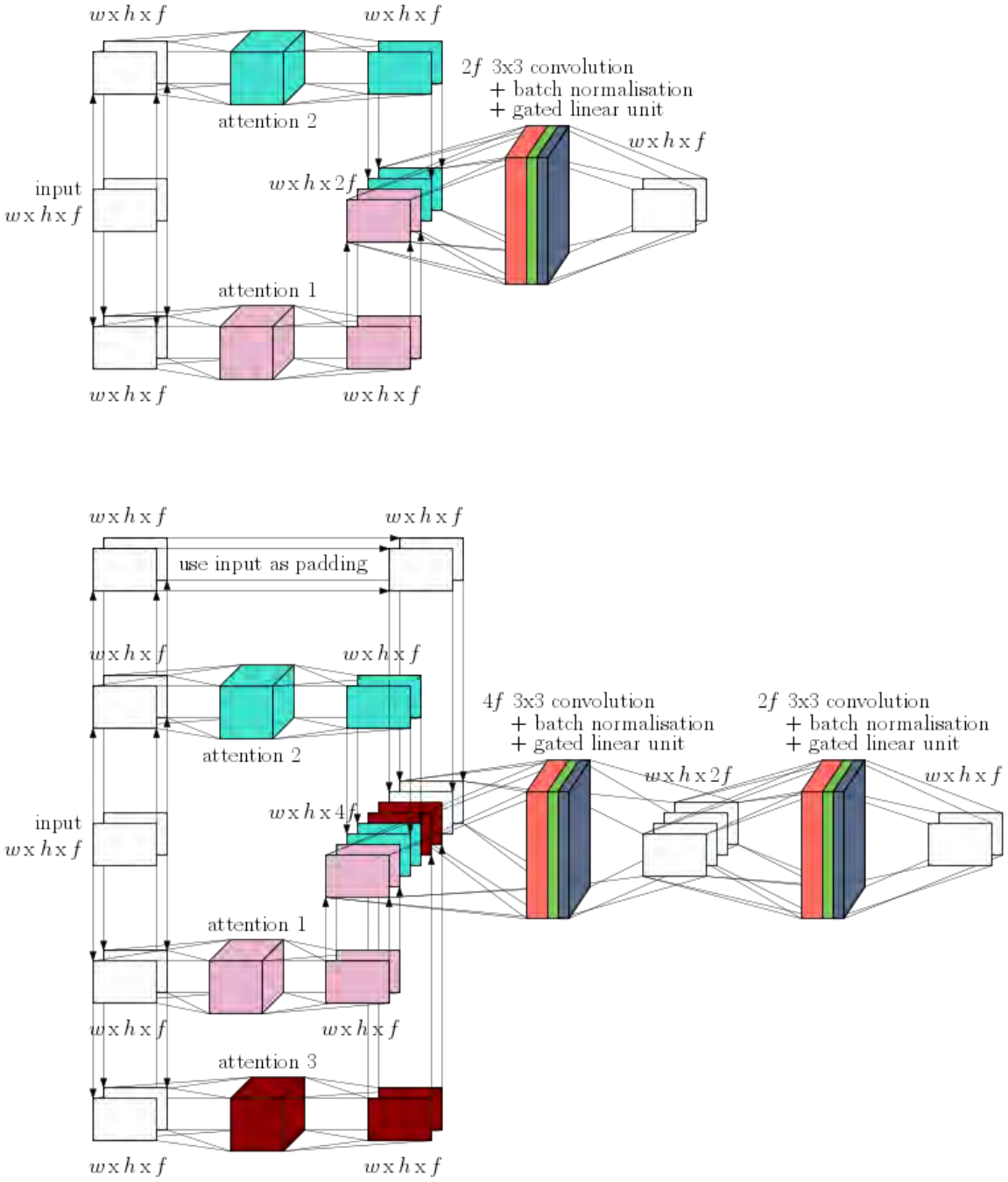

Figure 4.9.: Top: Combining two attention models using the gating property of the GLU. Bottom: Combining three attention models by using the original input as padding and two subsequent convolution+batch normalisation+GLU blocks. [9]

[9]Figure was created by author.

The attention module provides attention to each generator, except for the first (see Subsection 4.2.1). Because of the residual nature of the generators (see Figure 4.3), the attention for a generator is only computed once per epoch. This subsection illustrates how to combine multiple attention models with the pre-existing word attention. The techniques are generic for self-attention. Support for sentence attention is partially present.

Receiving feature maps as input, any of the self-attention models from Section 4.3 and word attention (not counting the additional word matrix input) yield a same-sized output. To maintain a direct residual path in the generators, the attention module has to follow that behaviour, i.e., the output size has to match the input size. Therefore, we condense the output of multiple attention models to the input size. We introduce two approaches: either using CBG which utilizes the gating property of the GLU or by viewing attention as a scalable heightmap and combining the heightmaps.

The CBG approach is simple and can be used for combining any number of attention models. First, if the number of attention models is not a power of two, then the input is used as padding (if needed multiple times) to achieve a power of two. Then, the appropriate number of convolution+batch normalisation+GLU blocks, hence CBG, is used until the output size matches the input size. Figure 4.9 depicts an example of two attention models and three attention models with the input as padding.

The second approach interprets attention as a scalable heightmap. Figure 4.10 illustrates an example of a feature map and a feature map modified by attention. Figure 4.11 demonstrates the concept of attention as a heightmap obtained by subtracting the original input from the attention-modified feature map.

Heightmaps of different attention models may contain values differing in magnitudes of order. In the CBG approach the convolution before the gating mechanism is mitigating that. Here, we scale each heightmap with a learnable parameter to facilitate a common order of magnitude.

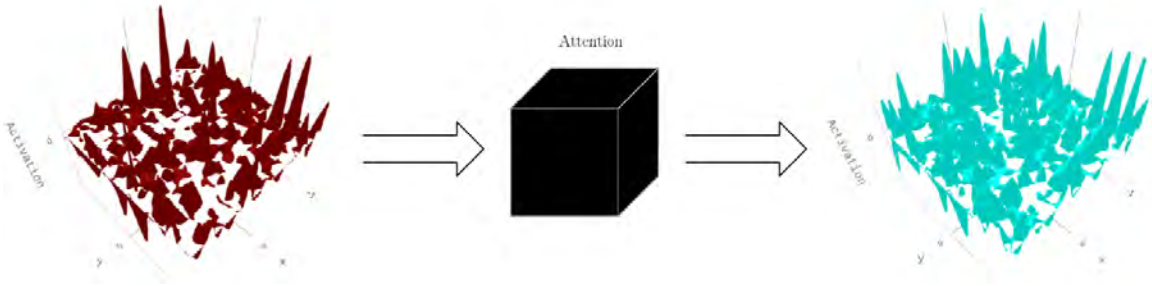

Figure 4.10.: Example of the positive side of a feature map and the positive side of a feature map after attention. Here, attention increased positive activity. [10]

[10] Figure was created by author.

Figure 4.11.: The left side shows a feature map (dark red) overlayed with its attention-modified version (light blue) (see Figure 4.10). Subtracting the unmodified version (dark red) from the attention-modified yields attention interpreted as a heightmap (right). Positive activations in the attention heightmap correspond to light blue surfaces in the left. Negative correspond to dark red surfaces. [11]

To combine multiple heightmaps, we propose using either a height-max or a mean operation. The height-max operation keeps the largest absolute value while preserving its original sign, see Figure 4.12. The intuition is that the bigger the value, i.e., the change in the original feature map, the more important this value is. However, it may be contraproductive if the attention models suggest changes in opposite directions.

The mean operation simply calculates the mean of the values, see Figure 4.12. The idea is that opposing values cancel each other out while values that the models agree upon remain. The mean operation tends to weaker responses, because the less confident model always dampens the response.

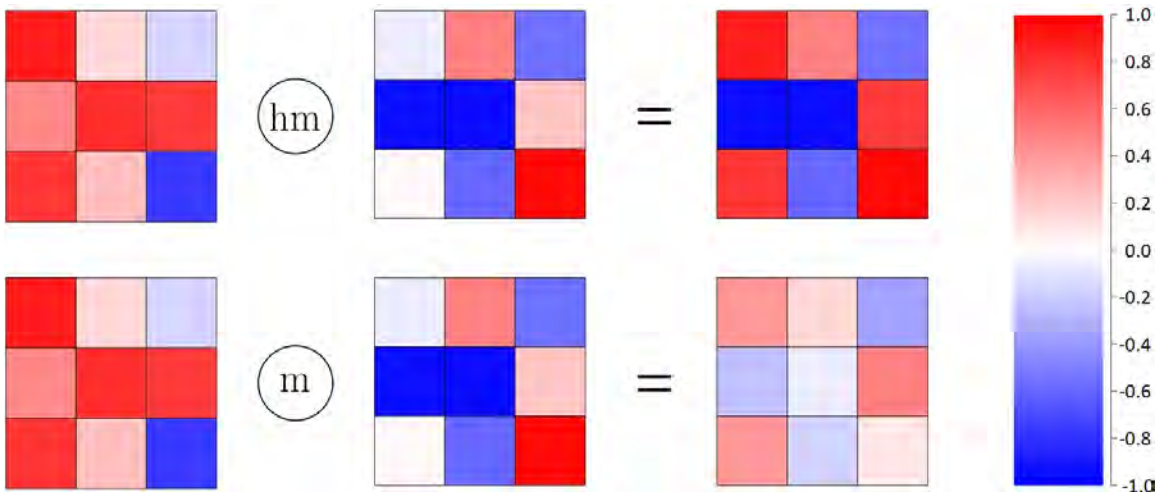

Figure 4.12.: Top: height-max operator: keeps the largest absolute value while preserving its original sign.
Bottom: mean operator: calculates the mean of the values. [12]

[11] Figure was created by author.

4.6. Spectral Normalisation (SN)

Miyato et al. introduced Spectral Normalisation (SN) [44] which is a weight normalisation technique to stabilise the training of the discriminator in GANs. It has shown to produce images of better or equal quality than previous training stabilisation techniques, such as weight normalisation [61], weight clipping [2], and gradient penalty [21]. Furthermore, the additional computational cost is small and the technique has only one low-impact hyperparameter to tune.

SN addresses common issues of GAN training such as mode collapse, vanishing gradients, and gradient explosion. Recent research indicates that the function space from which the discriminators are selected crucially affects the performance of GANs. A popular approach [21] [69] [51] is using Lipschitz continuity to assure the boundness of statistics. To do so, the Lipschitz constant of the discriminator is controlled by regularisation terms based on the input examples x. Spectral normalisation follows this approach by restraining the discriminator to a K-Lipschitz continuous function:

$$\arg \max_{\|f_{Lip}\| \leq K} V(G, D) \ . \tag{4.22}$$

$\|f_{Lip}\|$ is the smallest value M such that $\|f(x) - f(x')\| / \|x - x'\| \leq M$ for any x, x' with the norm being the l_2 norm. $f(x)$ is a simple discriminator made of a neural network with learning parameters $\theta := \{W^1, ..., W^{l+1}\}$.

For each layer $g : h_{in} \rightarrow h_{out}$ the Lipschitz norm $\|g\|_{Lip}$ is equal to $sup_h(\sigma(\nabla g(h)))$, where $\sigma(A)$ is the spectral norm of the matrix A (L_2 matrix norm of A):

$$\sigma(A) := \max_{h:h\neq 0} \frac{\|Ah\|_2}{\|h\|_2} = \max_{\|h\|_2 \leq 1} \|Ah\|_2 \ . \tag{4.23}$$

Therefore, for a linear layer $g(h) = Wh$ the Lipschitz norm is given by:

$$\|g\|_{Lip} = sup_h(\sigma(\nabla g(h))) = sup_h(\sigma(W)) = \sigma(W) \ . \tag{4.24}$$

If the Lipschitz norm of the activation function is equal to 1, then Equation 4.24 can be used to observe the following bound on f_{Lip} (for details we refer to [44]):

$$\|f_{Lip}\| \leq \prod_{l=1}^{L+1} \sigma(W^l) \ . \tag{4.25}$$

The spectral normalisation bounds the spectral norm of the weight matrix W so that it satisfies the Lipschitz constraint $\sigma(W) = 1$:

$$\overline{W}_{SN}(W) := W/\sigma(W) \ . \tag{4.26}$$

With W^l normalised per Equation 4.26 and $\sigma(\overline{W}_{SN}(W)) = 1$, Equation 4.25 bounds $\|f_{Lip}\|$ from above by 1. In conclusion, spectral normalisation bounds the Lipschitz norm, sets the spectral norm to a designated value, and augments the cost function with a sample data dependent regularisation function.

[12] Figure was created by author.

5. Experimental Results and Evaluation

This chapter illustrates our experiments, discusses evaluation metrics, and provides a comparison to state-of-the-art models. Section 5.1 illustrates the common setup of our experiments. Section 5.2 introduces several evaluation metrics such as the inception score (Subsection 5.2.1), wasserstein distance (Subsection 5.2.3), kernel maximum mean discrepancy (Subsection 5.2.4), 1-nearest neighbour classifier (Subsection 5.2.5), and the Fréchet inception distance (Subsection 5.2.2). Furthermore, it provides general thoughts on the suitability and use of the evaluation metrics and an in-depth discussion of the inception score. Lastly, Subsection 5.2.6 analyses the anti-correlation of the evaluation metrics and demonstrates that improvements on specific metrics, especially relative improvements, have to be viewed sceptically.

Section 5.3 outlines our experiments with different attention models, strategies to combine them, GAN training stabilising techniques, attention in the discriminator, and (partially) replacing convolutions with attention. In Subsection 5.3.7 we perform a hyperparameter tuning for our best models. A visual analysis of those tuned models follows in Subsection 5.3.8. Lastly, Section 5.4 compares our tuned models to state-of-the-art approaches and reinforces the need to use more than one evaluation metric.

5.1. Common Setup

This section illustrates the common setup of our experiments. We use the Caltech-UCSD Birds 200 (CUB) dataset [74], a well-known dataset for text-to-image generation consisting of 8855 train and 2933 test images. Each image has ten different captions. To compute our evaluation metrics, one image per caption in the test dataset is computed. Therefore, the evaluation metrics are computed over 29330 images. We compute our evaluation metrics every 25 epochs.

The images in the dataset are real-world images of 200 different classes of birds with varying backgrounds. The train/test split is oriented along the class of the bird: all bird images that show the same class of bird are either in the train or in the test split. The train split contains 150 classes and the test split 50 classes.

We use the Adam optimiser [33] for both the generators and the discriminators with a learning rate of 0.0002, $\beta_1 = 0.5$, and $\beta_2 = 0.999$. We train with a batch size of 20 because of the high memory consumption of images. We generate 256x256 images.

Each model employs spectral normalisation [44]. We follow the authors recommendation and use spectral normalisation with an l of 5. We use a λ of 5.0 because it performed best on our base architecture AttnGAN [81].

Unless explicitly stated otherwise, each model applies attention to the upsampling block according to Subsection 4.5.1 and to the attention module using the CBG method from Subsection 4.5.2 in conjunction with word attention.

Initially, we train each model for 400 epochs. Then, we train the best performing models for an additional 200 epochs. Furthermore, we perform a hyperparameter tuning of the common hyperparameters and model-specific hyperparameters on those models (see Subsection 5.3.7).

We evaluate each model along our five evaluation metrics. For the AttnGAN we use the officially reported inception score. We compute the other four evaluation metrics by evaluating the official model[1]. We indicate this by adding a * symbol to the AttnGAN in the legends.

Recent papers [86] [8] [10] using the AttnGAN as baseline have reported vastly different FID scores for the AttnGAN suggesting the use of different FID implementations. Therefore, a comparison to their FIDs, with the possible exception of [10], is futile (for details see Section 5.4 and Table 5.5).

5.2. Evaluation Metrics

Evaluating GANs is hard. Qualitative measures are inherently limited, subjective, time-consuming, and possibly misleading. Several quantitative metrics have been introduced, however, as of yet, there is no consensus as to which metric offers a fair model comparison. Furthermore, recent research [80] suggests that some of the metrics have serious limitations, including the inception score which, according to the authors of [4]: "fails to provide useful guidance when comparing models".

Moreover, the proposed metrics are solely for evaluating generative image models. They do not take into account the corresponding text in the context of text-to-image generation. Thus, they may be fooled by a network ignoring the textual input and only focusing on generating realistic looking images from the corresponding dataset.

We use several of the most popular evaluation metrics, namely the Inception Score (IS) (Subsection 5.2.1), Kernel Maximum Mean Discrepancy (MMD) (Subsection 5.2.4), the Wasserstein Distance (EMD) (Subsection 5.2.3), the 1-Nearest Neighbour Classifier (1-NN) (Subsection 5.2.5), and the Fréchet Inception Distance (FID) (Subsection 5.2.2).

[1] downloaded from https://drive.google.com/open?id=1lqNG75suOuR_8gjoEPYNp8VyT_ufPPig , see https://github.com/taoxugit/AttnGAN

Considering that the IS is perhaps the most widely adopted metric in text-to-image generation and used as a primary evaluation metric of our base architecture, we discuss the IS in detail. We also use it as our primary evaluation metric while keeping in mind the issues discussed in Subsection 5.2.1.

In Subsection 5.2.6 we analyse the anti-correlation of our evaluation metrics by searching for opposing responses. Our findings demonstrate that improvements on specific measures, especially relative improvements, have to be viewed sceptically.

The aim of a generative model is to use samples x to derive the unknown real data distribution $p_r(x)$. A generative model G encodes a distribution over new samples $p_g(x)$. The generative model aims to model the real data distribution as close as possible: $p_g(x) \sim p_r(x)$.

Unfortunately, GANs do not have an explicit representation of $p_g(x)$. Therefore, direct evaluation metrics, like the likelihood, cannot be used. This leads to many sampling-based metrics treating the generative model like a black-box: we assume that we can sample from $p_g(x)$ and assume nothing further of the structure of the model.

The feature space in which a metric is computed is of importance. The Inception v3 Network [66] (see Figure A.1) is a deep convolutional architecture designed for classification tasks on ImageNet [13]. ImageNet is a popular dataset consisting of 1.2 million RGB images from 1000 classes. We use the Inception v3 Network to compute our metrics, except the IS, in the convolutional (the output of the last inception model) feature space. Figure 5.1 visualizes the process. The IS uses the softmax (final output of the network) feature space (see Figure 5.2). All our evaluation metrics use the same pre-trained Inception v3 Network fine-tuned to the CUB dataset as our baseline the AttnGAN [81].

Using feature extractors before computing a metric's score may be misleading considering that the score may be unaffected by changes in the spatial relationship. However, it is still the preferred method as to directly comparing images, because there are many "correct" images for a textual description.

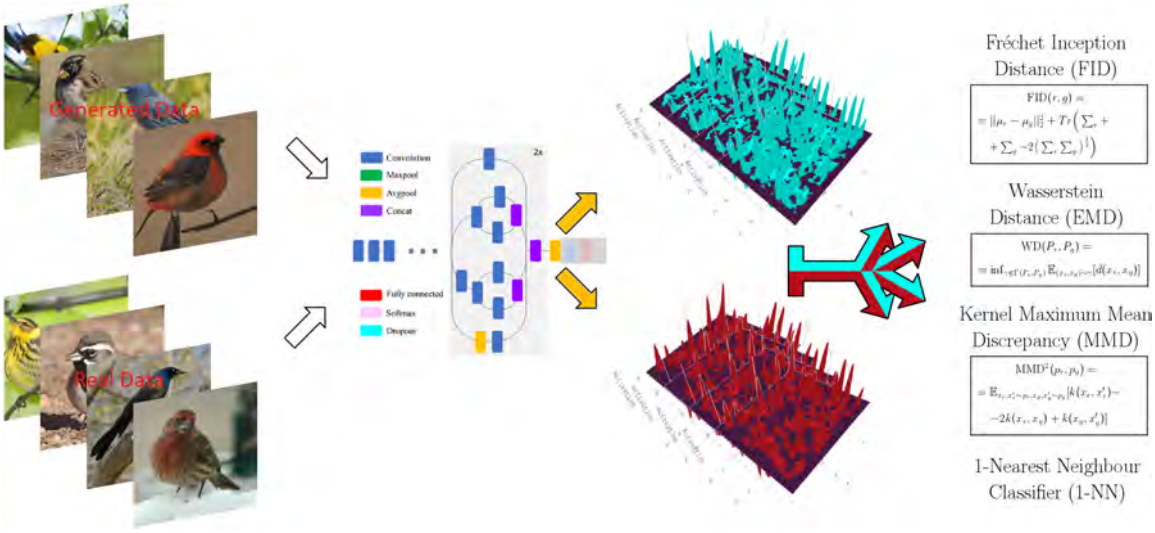

Figure 5.1.: Evaluation process for the FID, EMD, MMD, and 1-NN evaluation metrics. The full Inception v3 model is depicted in Figure A.1. [2]

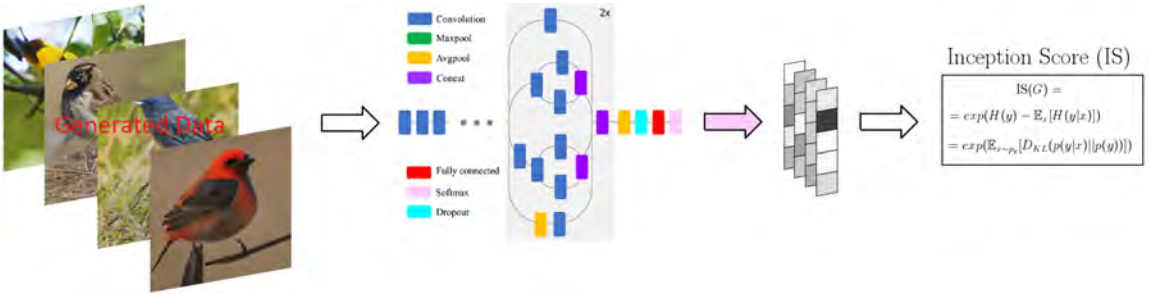

Figure 5.2.: Evaluation process for the IS evaluation metric. The full Inception v3 model is depicted in Figure A.1. [3]

5.2.1. Inception Score (IS)

The Inception Score (IS) [61] is a quantitative metric to evaluate generated images and is perhaps the most widely adopted score for text-to-image generation. It measures two properties: highly classifiable and diverse with respect to class labels. Salimans et al. introduced the IS and demonstrated a reasonable correlation between the IS and the quality and diversity of generated images. Figure 5.2 visualizes the evaluation process with the IS.

[2]Schematic Inception v3 model was altered from [43]. Rest of the figure was created by author.
[3]Schematic Inception v3 model was altered from [43]. Rest of the figure was created by author.

The IS for a network G measures the average Kullback-Leiber (KL) divergence between the conditional class distribution $p(y|x)$ and the marginal class distribution $p(y)$:

$$\mathrm{IS}(G) = exp(\mathbb{E}_{x\sim p_g}[D_{KL}(p(y|x)||p(y))]) = exp(H(y) - \mathbb{E}_x[H(y|x)]) \ . \tag{5.1}$$

$x \sim p_g$ indicates that x is sampled from p_g. $D_{KL}(p||q)$ is the KL-divergence between the distributions p and q. $H(x)$ represents entropy of the variable x. $p(y|x)$ is the conditional class distribution of the (generated) image x, estimated using the pretrained Inception v3 Network [66]. $p(y)$ is the marginal class distribution:

$$p(y) = \int_x p(y|x)p_g(x) \sim \frac{1}{N}\sum_{i=1}^{N} p(y|x_i)p_g(x_i) \ . \tag{5.2}$$

$p(y|x)$ is expected to have low entropy for better classifiable samples. Therefore, it is supposed to encourage a better sample quality. $p(y)$ is expected to have high entropy if all classes are equally represented in the set of samples. Thus, it is supposed to encourage high diversity.

Although the IS is perhaps the most widely used metric in text-to-image generation, it has several issues regarding the computation of the score itself and the usage of the score.

Rosca et al. [60] point out that the IS was originally proposed for generative image models trained on ImageNet. Therefore, the use of the IS on other datasets may be misleading. For example, a simple class-conditional model memorising one example per ImageNet class achieves a high IS.

Furthermore, Barratt and Sharma [4] demonstrate with a simple one-dimensional example that the true underlying data distribution may achieve a lower IS than other distributions. They also raise another significant issue: while the classification accuracy of the inception network (v2 or v3) is robust against slight weight changes, the IS itself is not. They demonstrate this behaviour by showing that the IS varies up to 11.5%, depending on whether a tensorflow, keras, or pytorch implementation of the inception network with virtually the same classification accuracy is used.

To avoid this specific issue, we use the same IS implementation, using an Inception v3 Network fine-tuned to the CUB dataset, as our baseline the AttnGAN [81]. Lastly, Odena et al. [45] show that the IS is asymmetric and affected by image resolution.

5.2.2. Fréchet Inception Distance (FID)

Heus et al. introduced the Fréchet Inception Distance (FID) [24]. The FID is computed in the convolutional feature space of the pretrained Inception v3 Network (see Figure 5.1 for the evaluation process with the FID). The features are viewed as a continuous multivariate Gaussian and the mean μ and the covariance $\sum$ are computed for both the real data r and

the generated data g. With these the FID computes as:

$$\text{FID}(r, g) = ||\mu_r - \mu_g||_2^2 + Tr\Big(\sum_r + \sum_g -2\big(\sum_r \sum_g\big)^{\frac{1}{2}}\Big) \ . \tag{5.3}$$

A lower FID implies a closer distance between the generated image distribution and the real image distribution. The FID is consistent with human judgment and more consistent to noise than the IS [24]. Furthermore, it is able to detect a model generating only one sample per class which still scores a high IS (see Subsection 5.2.1) but a bad FID score. However, the FID assumes that features are of Gaussian distribution which is often not the case. Moreover, Lucic et al. [40] show that the FID has a slight bias.

We use the official pytorch implementation[4]of the FID. To ensure a consistent calculation of all of our evaluation metrics, we replace the generic Inception v3 network with the pre-trained Inception v3 Network fine-tuned to the CUB dataset used by all our other evaluation metrics and for the AttnGAN [81].

5.2.3. Wasserstein Distance (EMD)

The wasserstein distance measures the minimum mass displacement to transform one distribution into the other. The closer the distributions, the smaller the distance. For two distributions P_r and P_g it is defined as:

$$\text{WD}(P_r, P_g) = \inf_{\gamma \in \Gamma(P_r, P_g)} \mathbb{E}_{(x_r, x_g)\sim\gamma}[d(x_r, x_g)] \ . \tag{5.4}$$

$\Gamma(P_r, P_g)$ denotes the set of all joint distributions whose marginals are P_r and P_g, respectively. $d(x, y)$ denotes the base distance between two samples. For discrete distributions p_r and p_g the wasserstein distance is often referred to as the Earth Mover's Distance (EMD) and corresponds to the solution of the optimal transport problem:

$$\begin{aligned} \text{EMD}(p_r, p_g) &= \min_{w \in \mathbb{R}^{nxm}} \sum_{i=1}^{n} \sum_{j=1}^{m} w_{ij} d(x_r^i, x_g^j) \\ \text{subject to } \sum_{j=1}^{m} w_{ij} &= p_r(x_r^i) \ \forall i, \sum_{i=1}^{n} w_{ij} = p_g(x_g^j) \ \forall j \ . \end{aligned} \tag{5.5}$$

This finite sample approximation of the wasserstein distance is used in practice. Figure 5.1 visualizes the evaluation process with the EMD.

5.2.4. Kernel Maximum Mean Discrepancy (MMD)

Kernel Maximum Mean Discrepancy measures the dissimilarity of two distributions p_r and p_g for some fixed kernel function k:

$$\text{MMD}^2(p_r, p_g) = \mathbb{E}_{x_r, x_r' \sim p_r, x_g, x_g' \sim p_g}[k(x_r, x_r') - 2k(x_r, x_g) + k(x_g, x_g')] \ . \tag{5.6}$$

[4]https://github.com/bioinf-jku/TTUR

A lower MMD implies a closer distance between the two distributions. In practice, finite samples from distributions are used to estimate the MMD. Therefore, the MMD may not be zero even if the distributions are identical. Figure 5.1 visualizes the evaluation process with the MMD.

5.2.5. 1-Nearest Neighbour Classifier (1-NN)

The 1-Nearest Neighbour Classifier (1-NN) is part of the two-sample test family. It tests whether two distributions are identical. Given two sets of samples $s_r \sim p_r$ and $s_g \sim p_g$, the Leave-One-Out (LOO) accuracy of a 1-NN classifier trained on s_r and s_g is computed. s_r uses positive labels and s_g uses negative labels. While any binary classifier can be used to compute the LOO accuracy, the 1-NN classifier is convenient because it requires no special training and little hyperparameter tuning.

Ideally, the LOO accuracy is 50%. If the GAN memorizes every sample in s_r and regenerates them perfectly, then every sample from s_g has its nearest neighbour in s_r with a distance of zero resulting in a 0% LOO accuracy. [80] Figure 5.1 visualizes the evaluation process with the 1-NN.

5.2.6. Anti-Correlation of Evaluation Metrics

This subsection analyses the anti-correlation among our evaluation metrics. Section 5.2 establishes that some of the metrics have serious limitations, especially the IS, and that there is no consensus as to which metric offers a fair model comparison. We concur with this statement by demonstrating anti-correlation, both relative and normalised, among our evaluation metrics.

For that we search for opposing responses of our evaluation metrics across all our experiments. For an evaluation metric e, two models m_1 and m_2, and a specific epoch t we define the relative improvement r as:

$$r = \begin{cases} e(m_1, t)/e(m_2, t) - 1.0, & \text{if } e = \text{IS} \\ -(e(m_1, t)/e(m_2, t) - 1.0), & \text{otherwise .} \end{cases} \tag{5.7}$$

We need to differentiate between the IS and the other evaluation metrics because a higher IS is a positive improvement whereas for the other evaluation metrics a lower score is a positive improvement. For two evaluation metrics e_a and e_b we search for two models m_1 and m_2 and a specific epoch t such that r_a and r_b yield opposing responses, i.e., we search for occurrences of a relative improvement on one metric with a relative deterioration on the other. We define the relative anti-correlation as the sum of the absolutes of two opposing relative improvements:

$$|r_a| + |r_b| \quad \text{, where} \quad r_a \geq 0,\ r_b < 0 \quad \text{or} \quad r_a < 0,\ r_b \geq 0\ . \tag{5.8}$$

We visualize this process in Figure 5.3. There, we mainly observe anti-correlation between the IS and the other evaluation metrics: the IS displays a positive relative improvement

whereas the other metrics display a negative relative improvement. At epoch 225 and 250 the MMD displays anti-correlation to the EMD, 1-NN, and FID. This demonstrates that the evaluation metrics may vary significantly in their response and that a relative improvement on a specific metric is not necessarily meaningful.

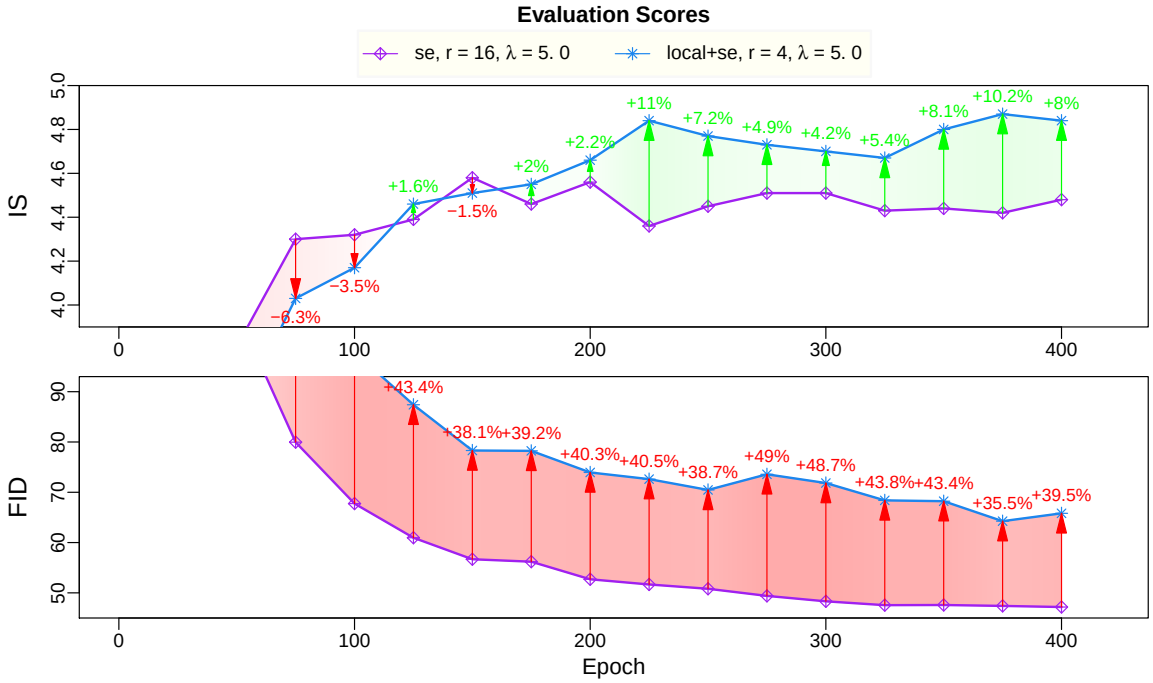

Figure 5.3.: Relative improvements on the IS and FID (see Figure A.2 for EMD, MMD, and 1-NN) of local self-attention with se attention over se attention. [5]

Maximising the relative anti-correlation for each pair of evaluation metrics results in Table 5.1. The first 49 epochs are excluded because of the initial rapid convergence of the evaluation metrics (see Table 5.2). Moreover, both local self-attention in the discriminator and replacing convolutions with local self-attention are excluded due to failure states during training. The relative anti-correlation is symmetrical per definition. Each row highlights its maximum. The IS is the only metric with an uncertainty. Therefore, only relative anti-correlation involving the IS has an uncertainty.

We observe that the FID and the IS have the strongest overall relative anti-correlation while the EMD has the lowest. In the case of the 81% ± 2% between the IS and the FID the IS went from 4.28 to 4.33 yielding a positive improvement of +1.2% while the FID went from 52.27 to 94.07 yielding a negative improvement of −80.0%. This demonstrates that using a single evaluation metric can be misleading.

[5] Figure was created by author.

Table 5.1.: Maximum relative anti-correlations of our evaluation metrics in our experiments excluding the first 49 epochs and models with failure states during training.

	IS	FID	EMD	MMD	1-NN
IS	-	**81% ± 2%**	23% ± 3%	26% ± 2%	23% ± 3%
FID	**81% ± 2%**	-	13%	52%	45%
EMD	**23% ± 3%**	13%	-	21%	13%
MMD	26% ± 2%	**52%**	21%	-	10%
1-NN	23% ± 3%	**45%**	13%	10%	-

While our findings on relative anti-correlation yield insights into the significance of relative improvements, they are not an assessment of the quality of the evaluation metrics, because they are inherently different. We can observe that by examining their value ranges (see Table 5.2): excluding the first 49 epochs the IS ranges between 3.37 and 4.96 in our experiments, whereas the FID ranges between 137.14 and 42.49. Thus, the maximum relative improvement of the IS is 47%, whereas of the FID it is −223%. For the 1-NN score the maximum relative improvement is at −6%. As a consequence, the relative anti-correlation of the 1-NN and any other evaluation metric consists mainly of the other metrics relative improvement.

Moreover, the same difference in value yields different positive and negative improvements. In the case of the 81% ± 2% between the IS and the FID changing the point of view to a deterioration of the IS from 4.33 to 4.28 and an improvement of the FID from 94.07 to 52.27 results in a relative anti-correlation of 46%.

Table 5.2.: Occurring value ranges of our evaluation metrics. min (50+) and max (50+) exclude the first 49 epochs. In addition, both local self-attention in the discriminator and replacing convolutions with local self-attention are excluded due to failure states during training.

	IS↑	FID↓	EMD↓	MMD↓	1-NN$^{\downarrow}_{\uparrow}$
min	1.00	399.62	18.71	0.705	1.000
min (50+)	3.37	137.14	15.05	0.209	0.994
max	4.96	42.49	11.36	0.141	0.936
max (50+)	4.96	42.49	11.36	0.141	0.936

To gain further insights into the behaviour of the evaluation metrics we examine normalised anti-correlation. The idea is to normalise each evaluation metric using the range of their occurring values. The first 49 epochs are excluded because of the initial rapid convergence of the evaluation metrics (see Table 5.2). This may be misleading if a technique has a major impact on one evaluation metric but not on the others. However, the evaluation metrics are not bound, except for the 1-NN, thereby we have to introduce bounds. Furthermore, this allows to observe the strength of the reaction of the evaluation metrics in regards to each other.

For an evaluation metric e, two models m_1 and m_2, and a specific epoch t we define the normalised improvement n as:

$$n = \begin{cases} (e(m_1,t) - e(m_2,t))/(e^{max50+} - e^{min50+}), & \text{if } e = \text{IS} \\ -(e(m_1,t) - e(m_2,t))/(e^{max50+} - e^{min50+}), & \text{otherwise} , \end{cases} \tag{5.9}$$

where $[e^{min50+}, e^{max50+}]$ is the range of the occurring values excluding the first 49 epochs for e. The values of n range between -1 and 1, where -1 represents the worst occurring negative improvement from e^{max50+} to e^{min50+} and 1 the opposite. The values in between represent a linear interpolation: a value x represents an improvement of $x \cdot (e^{max50+} - e^{min50+})$. We define the normalised anti-correlation as the mean of the absolutes of two opposing normalised improvements:

$$\frac{|n_a| + |n_b|}{2} \quad \text{, where} \quad n_a \geq 0,\ n_b < 0 \quad \text{or} \quad n_a < 0,\ n_b \geq 0 \ . \tag{5.10}$$

Maximising the normalised anti-correlation for each pair of evaluation metrics results in Table 5.3. The first 49 epochs are excluded because of the initial rapid convergence of the evaluation metrics (see Table 5.2). The normalised anti-correlation is symmetrical per definition. Its values range between 0 and a 1. Moreover, the same difference yields the same positive and negative improvement. Each row highlights its maximum. The IS is the only measure with an uncertainty. Therefore, only normalised anti-correlation involving the IS has an uncertainty.

Table 5.3.: Maximum normalised anti-correlations of our evaluation metrics in our experiments excluding the first 49 epochs and models with failure states during training.

	IS	FID	EMD	MMD	1-NN
IS	-	31% ± 4%	34% ± 3%	32% ± 3%	**41% ± 3%**
FID	**31% ± 4%**	-	6%	20%	21%
EMD	**34% ± 3%**	6%	-	28%	28%
MMD	**32% ± 3%**	20%	28%	-	19%
1-NN	**41% ± 3%**	21%	28%	19%	-

We observe that the IS has the largest normalised anti-correlation. This behaviour most likely originates from its several issues discussed in Subsection 5.2.1. The EMD and FID have the lowest normalised anti-correlation of 6% and second-lowest relative anti-correlation of 13%. This concurs with strongly coherent behaviour of both evaluation metrics throughout our experiments. In general, our findings demonstrate that improvements on a specific evaluation metric have to be viewed sceptically, especially relative improvements.

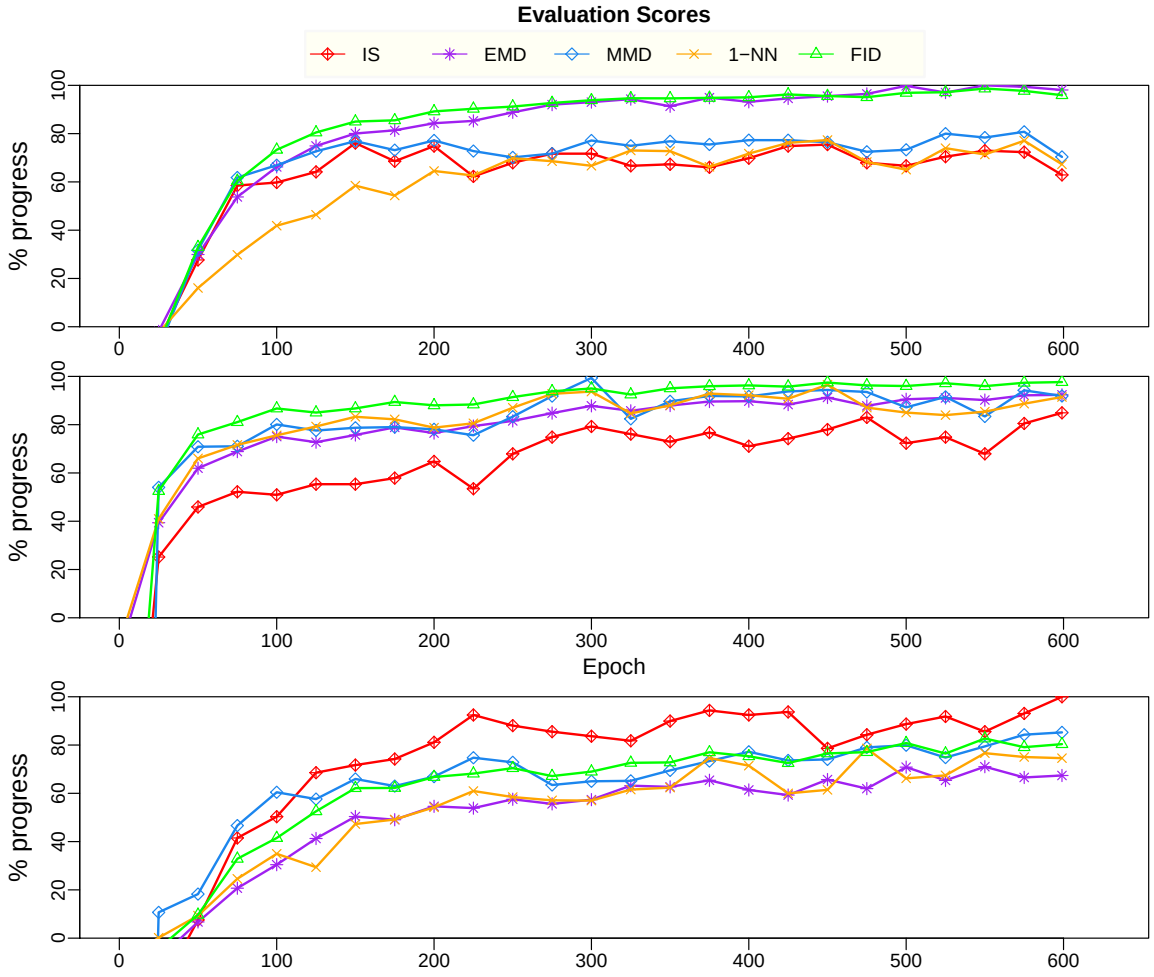

Figure 5.4.: Normalised evaluation metrics for se attention with $\lambda = 5.0, r = 16$ (top), with $\lambda = 0.1, r = 16$ (middle), and se attention with local self-attention with $\lambda = 5.0, r = 4$. [6]

Lastly, normalising the evaluation metrics allows us to directly compare them and to observe the strength of their responses. In Figure 5.4 we see all our evaluation metrics normalised for three similar models, all displaying a different behaviour. The first se attention model (top) shows two groups of response strengths: the EMD and the FID with near identical maximum response strengths and the other three evaluation metrics with a clearly weaker response ranging between 60% and 80%.

The second se attention model (middle) demonstrates that a little alteration, such as lowering the hyperparameter λ from 5.0 to 0.1, may have a huge impact on some evaluation metrics. We observe a major positive impact on the MMD and 1-NN with the MMD showing a maximum response at epoch 300. The IS also displays a positive impact

[6] Figure was created by author.

now reaching above 80%. The FID remains near its maximum response. Only the EMD shows a negative response to this hyperparameter-tuning performing 5% to 10% lower as previously. Overall, all five evaluation metrics show a strong correlation, especially the four without the IS.

In the previous two models the IS had the lowest performance. In the third model the IS responds the strongest, reaching its maximum response strength at epoch 599. The EMD displays the weakest response of about 60%. The third model uses local self-attention. In addition, it uses se attention after every convolution, with the exception of the discriminator and convolutions used in attention mechanisms. Furthermore, the internal bottleneck reduction hyperparameter r is lowered from 16 to 4.

In conclusion, our examples demonstrate that evaluation metrics may react significantly different to different forms of attention and may even show strong reactions to little alterations, such as hyperparameter tuning. Therefore, results on single evaluation metrics must be viewed sceptically. They also indicate that the IS behaves most erratically out of our five evaluation metrics concurring with our analysis of relative and normalised anti-correlation.

5.3. Models

This section outlines our experiments with different attention models, strategies to combine them, attention in the discriminator, and (partially) replacing convolutions with attention. Section 5.1 outlines the common setup of our experiments and the definitions of hyperparameters and abbreviations.

5.3.1. Global vs. Local Self-Attention

Figure 5.5 shows the impact of global and local self-attention. Shown are 5 models: global, local, spatially-aware local, and global self-attention mixed with local/spatially-aware local self-attention. When mixing, we initially use global self-attention and then switch to local self-attention if a spatial dimension of the input of the layer is ≥ 128. For global self-attention spatial dimensions ≥ 128 of the input are downsampled to 64 to avoid large memory consumption (see Subsection 4.3.1).

Global and global self-attention mixed with either local models behave similar, because the network mostly relies on global self-attention. Only in the second generator is the input large enough in the attention module to switch to local self-attention. However, we would expect a significant difference for larger images due to the down- and upsampling to and from 64, whereas local self-attention is mostly spatially independent.

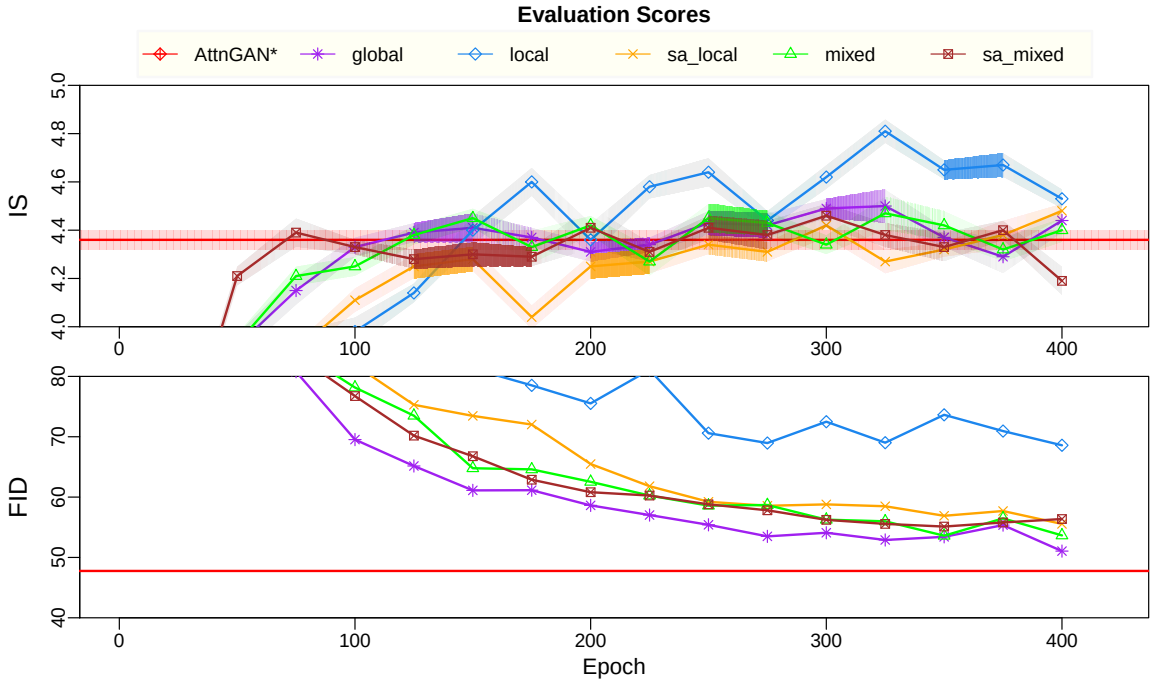

Figure 5.5.: IS and FID (see Figure A.3 for EMD, MMD, and 1-NN) of global, local, spatially-aware local, and global self-attention mixed with local/spatially-aware local self-attention. When mixing, local self-attention is used if a spatial dimension of the input is ≥ 128. Otherwise, the input is downsampled to 64 for global self-attention (see Subsection 4.3.1). [7]

We observe that each model reaches the uncertainty region of the IS of the AttnGAN. However, only local self-attention shows significant improvements boosting the IS by 10.3% ± 2.2% from 4.36 ± 0.04 to 4.81 ± 0.05 at epoch 325. In contrast, local self-attention displays major negative improvements on the FID score. At epoch 325 the FID increases by 44.5% from 47.76 to 69.01. The EMD reflects this behaviour displaying major negative improvements as well. Both the MMD and the 1-NN show similar scores on all models. These significantly different responses of the evaluation metrics are discussed in Subsection 5.2.6. In addition, due to its excellent performance on the IS the local self-attention model receives further training and hyperparameter tuning in Subsection 5.3.7.

[7] Figure was created by author.

5.3.2. Sentence Attention

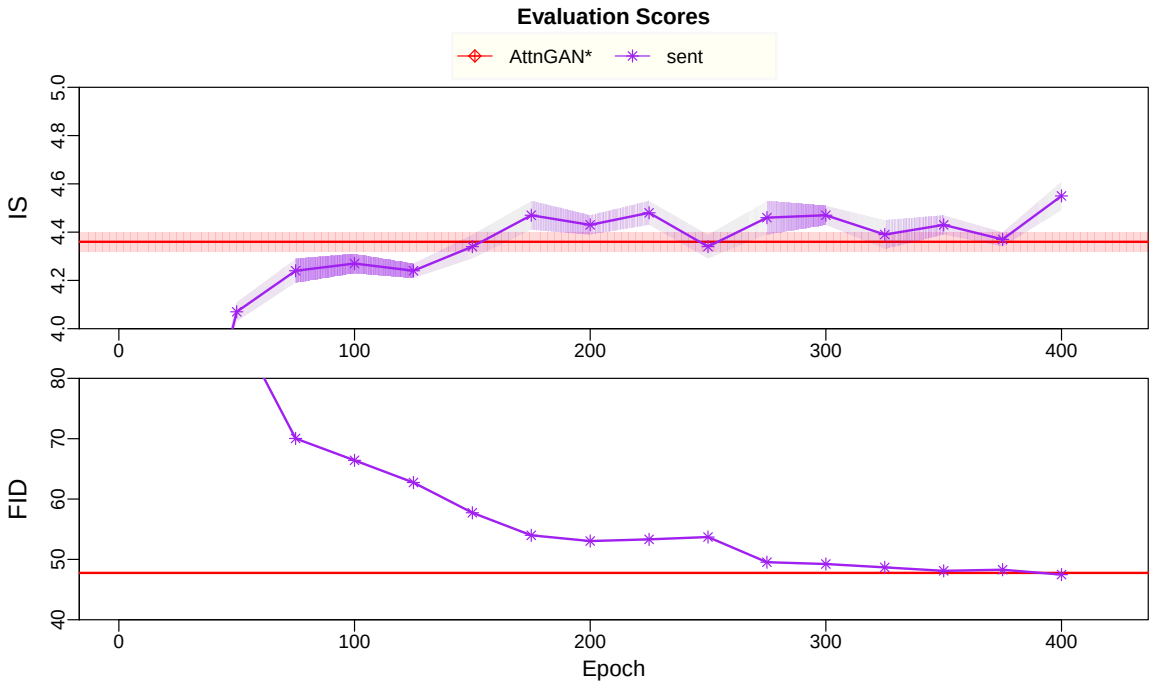

Figure 5.6.: IS and FID (see Figure A.4 for EMD, MMD, and 1-NN) of sentence attention. [8]

Figure 5.6 compares sentence attention to the AttnGAN. Sentence attention is only applied to the attention model using the CBG method from Subsection 4.5.2 in conjunction with word attention. Unlike the other attention models, it is not applied to upsampling blocks.

It shows that the results remain largely unaffected by sentence attention. The minor improvements on some of the evaluation metrics may originate from spectral normalisation. We trace this result back to word attention already encompassing the important information of the sentence in a more fine-grained matter. Therefore, our results endorse the effectiveness of the pre-existing word attention mechanism and show that additional sentence attention is obsolete.

[8] Figure was created by author.

5.3.3. Squeeze-And-Excitation Attention

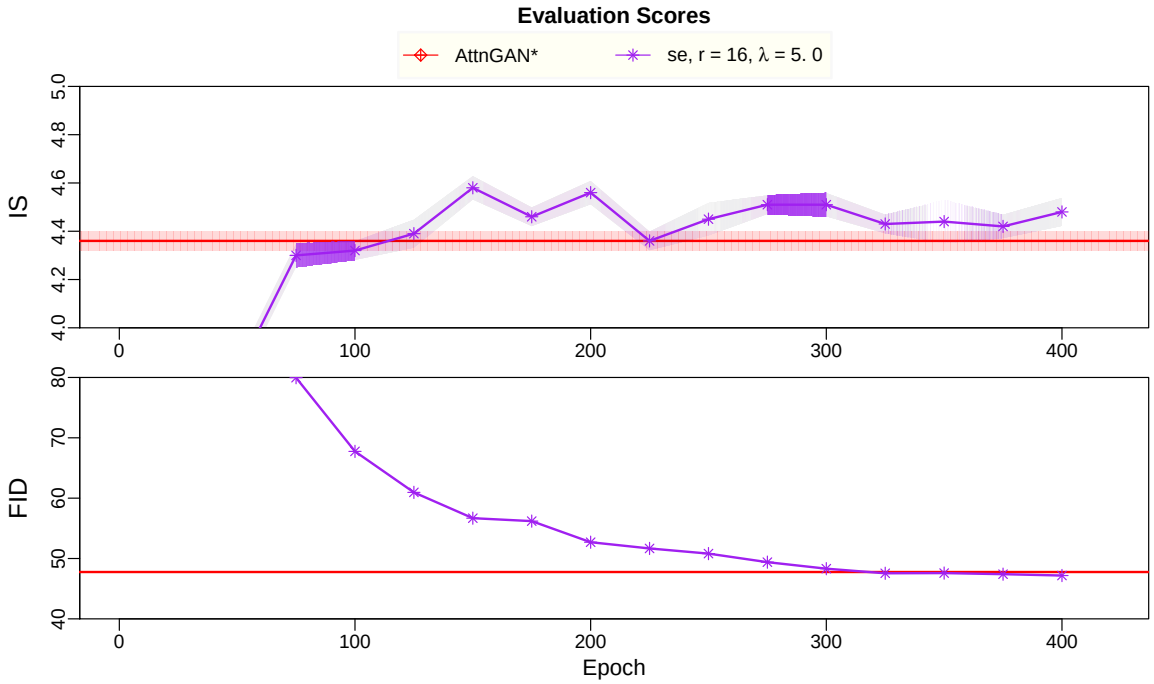

Figure 5.7.: IS and FID (see Figure A.5 for EMD, MMD, and 1-NN) of squeeze-and-excitation attention after every convolution (with the exception of the discriminator and convolutions used in attention). [9]

Figure 5.7 compares squeeze-and-excitation attention to our baseline. Unlike self-attention, squeeze-and-excitation attention is applied after every convolution, with the exception of the discriminator and convolutions used in attention mechanisms. We observe minor improvements on the IS, MMD, and 1-NN and comparable results on the EMD and FID. Therefore, the se model receives further training and hyperparameter tuning in Subsection 5.3.7.

[9] Figure was created by author.

5.3.4. Combining Attention Models

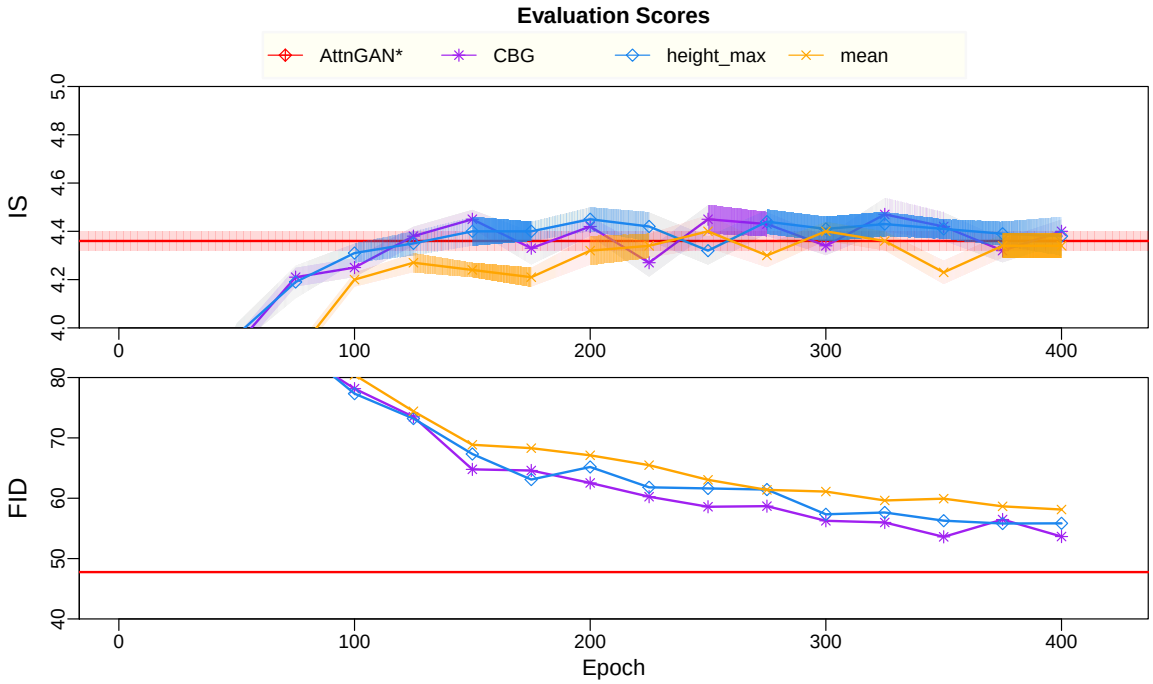

Figure 5.8.: IS and FID (see Figure A.6 for EMD, MMD, and 1-NN) of global mixed with local self-attention combined using the convolution+batch normalisation+GLU (CBG) approach or by viewing attention as heightmaps and using the height-max or mean approach. [10]

Figure 5.8 investigates different techniques of combining attention. It shows that the different techniques of combining attention only have a mild impact. The height_max approach shows slightly better results on the IS but slightly worse results on the EMD and FID than the CBG approach. The mean approach displays slightly inferior or comparable results across all evaluation metrics.

As stated in Section 5.1 we chose the CBG approach for our other models. Compared to the other two approaches, the CBG approach is able to consider the neighbourhood of a value, does not require individual scaling, and has learnable parameters to adapt to each task. However, our results show that the height_max approach is also a viable option, especially when focusing on the IS. In addition, one of the other approaches may harmonise better with a different attention model than global mixed with local self-attention.

[10] Figure was created by author.

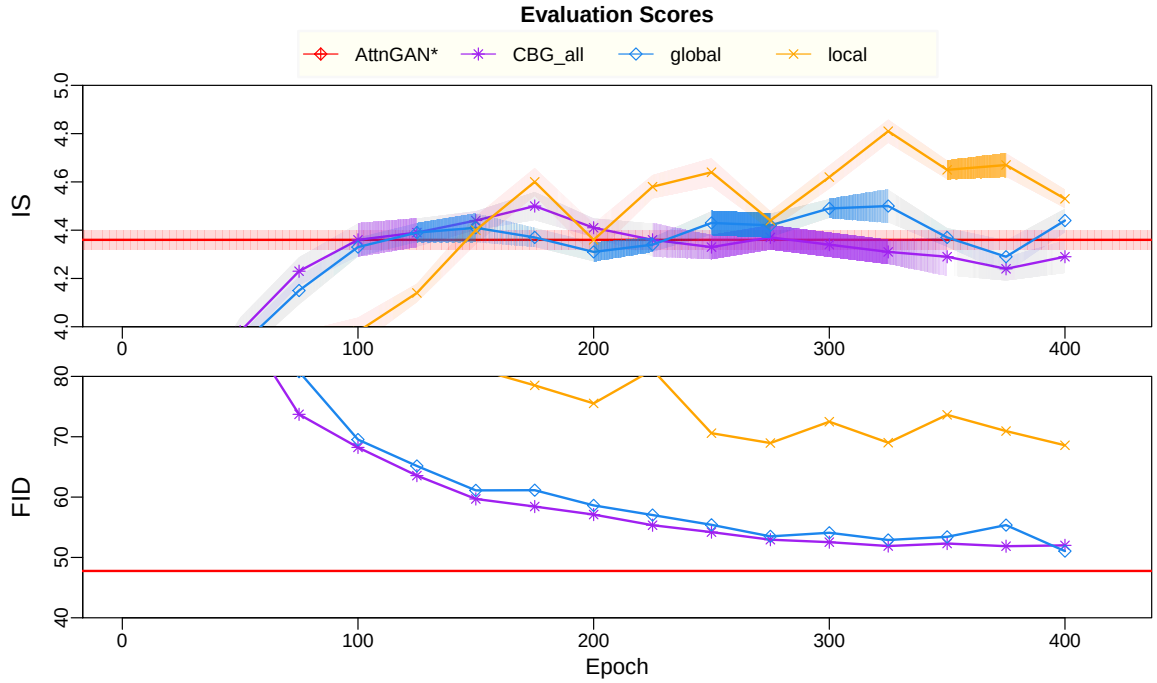

Figure 5.9.: IS and FID (see Figure A.7 for EMD, MMD, and 1-NN) of combining global, local, and spatially aware local self-attention with word attention using the CBG method (CBG_all) and of global and local self-attention. [11]

Figure 5.9 displays the effect of combining global, local, and spatially aware local self-attention with word attention using the CBG method. Initially, we observe a spike in the IS indicating a strong influence of local self-attention. This behaviour is to be expected, because global self-attention starts with a γ of zero and then gradually assigns more importance to global self-attention. As the training progresses all evaluation metrics approximate the results of global self-attention.

Therefore, global and local self-attention should be used on its own or require regulation, such as a constant γ, to prevent one attention model from dominating the other(-s).

[11] Figure was created by author.

5.3.5. Attention in the Discriminator

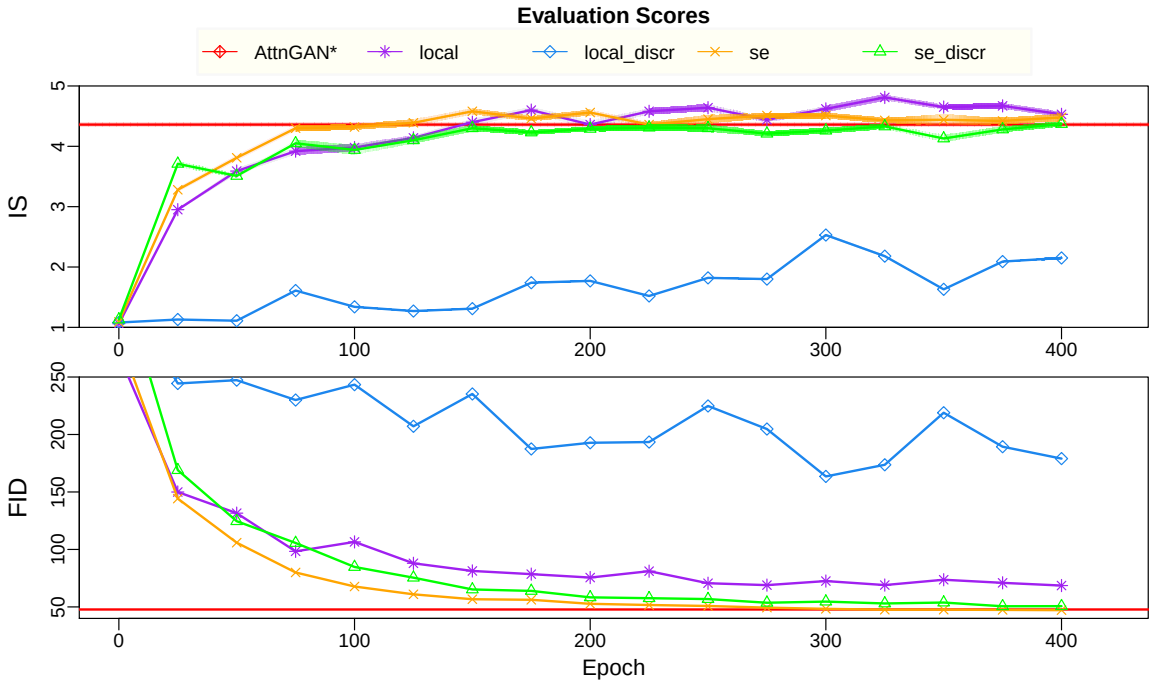

Figure 5.10.: IS and FID (see Figure A.7 for EMD, MMD, and 1-NN) of local self-attention and of adding local self-attention in the discriminators. [12]

So far, all proposed techniques aimed to enhance the generator. We experiment with using local self-attention before every convolution in the discriminator and with using se attention after every convolution in the discriminator. We refrain from from the use of global self-attention due to its memory intensive nature for large images.

Figure 5.10 illustrates that se attention in the discriminator has no major impact. The model behaves similar to our normal se model. However, it performs slightly worse across all evaluation metrics. Local self-attention in the discriminator behaves chaotically and learns very slowly. On the 1-NN score it displays no learning at all.

We assume that the min-max game of the GAN is impaired when using local self-attention. Figure 5.11 visualizes the training errors of each generator and discriminator and the DAMSM loss. The large jumps in all three, especially in the second discriminator and generator, indicate mode collapse. This may originate from enhanced capabilities of the discriminators causing it to learn too fast and impairing the min-max GAN game. Adjusting the learning rates may solve this issue. Alternatively, local self-attention may be unsuited for the discriminator task at hand.

[12] Figure was created by author.

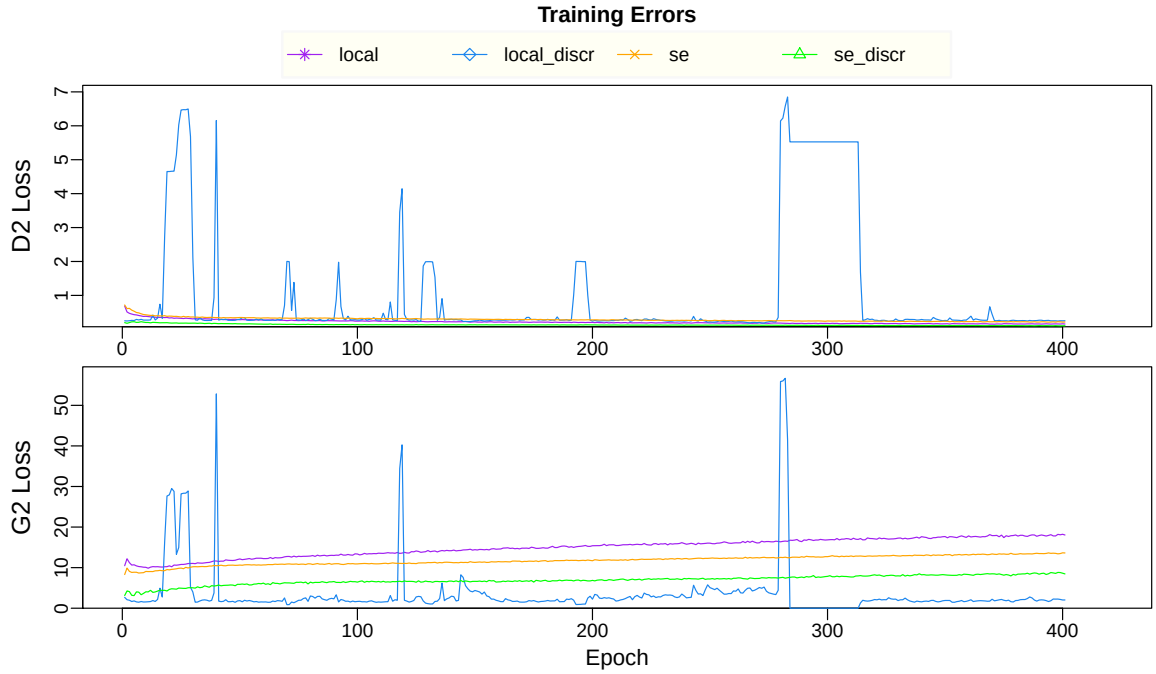

Figure 5.11.: Training losses of discriminator 2 and generator 2 (for 0, 1, and the DAMSM loss see Figure A.10) of local self-attention and of adding local self-attention in the discriminator. [13]

5.3.6. Replacing Convolutions

We experiment with replacing convolutions, excluding 1x1 convolutions, in the generators with local self-attention. We refrain from the use of global self-attention due to its computational cost for large images.

Figure 5.12 shows that replacing convolutions does not learn across all evaluation metrics. However, analysing the training losses (Figure 5.13) reveals that the network does learn and improves upon its loss functions. The discriminators perform slightly better while the generator losses are uncharacteristically high for their accompanying discriminator losses based on the behaviour of our other models. The high DAMSM loss suggests that the network is less susceptible to the DAMSM. A higher λ may resolve this issue and yield competitive results.

Alternatively, the min-max game of the GAN may be impaired; or local self-attention alone may not be suited for the generative task at hand; or the used loss functions may be misleading.

[13] Figure was created by author.

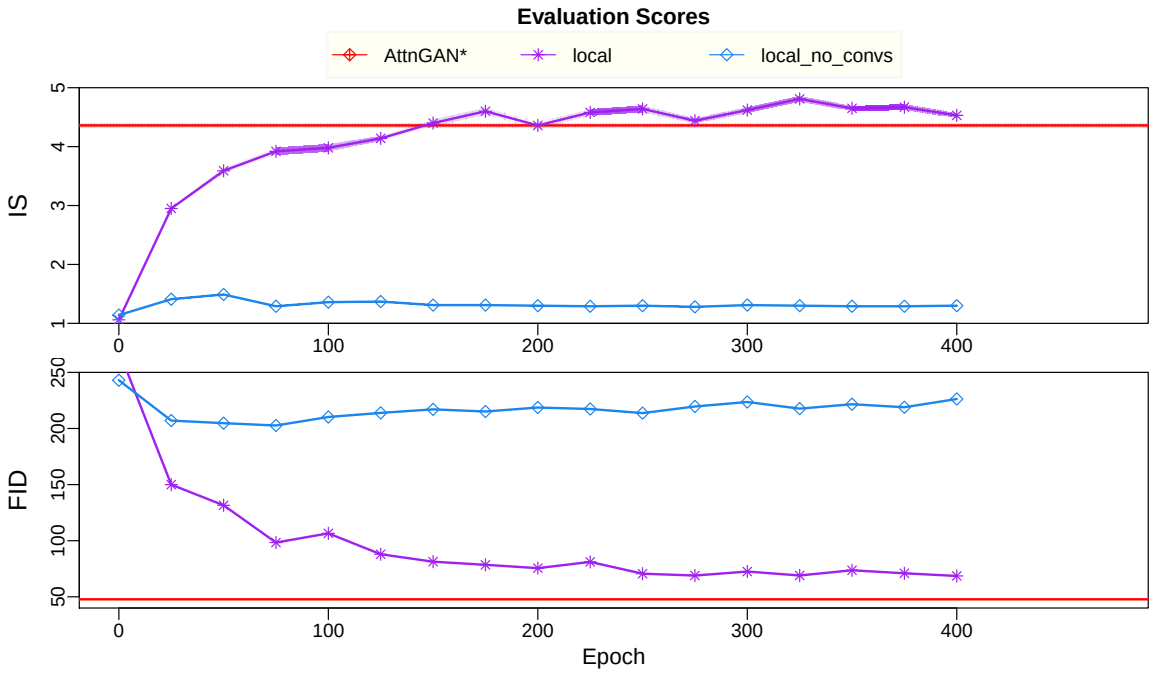

Figure 5.12.: IS and FID (see Figure A.9 for EMD, MMD, and 1-NN) of replacing convolutions in the generators with local self-attention. [14]

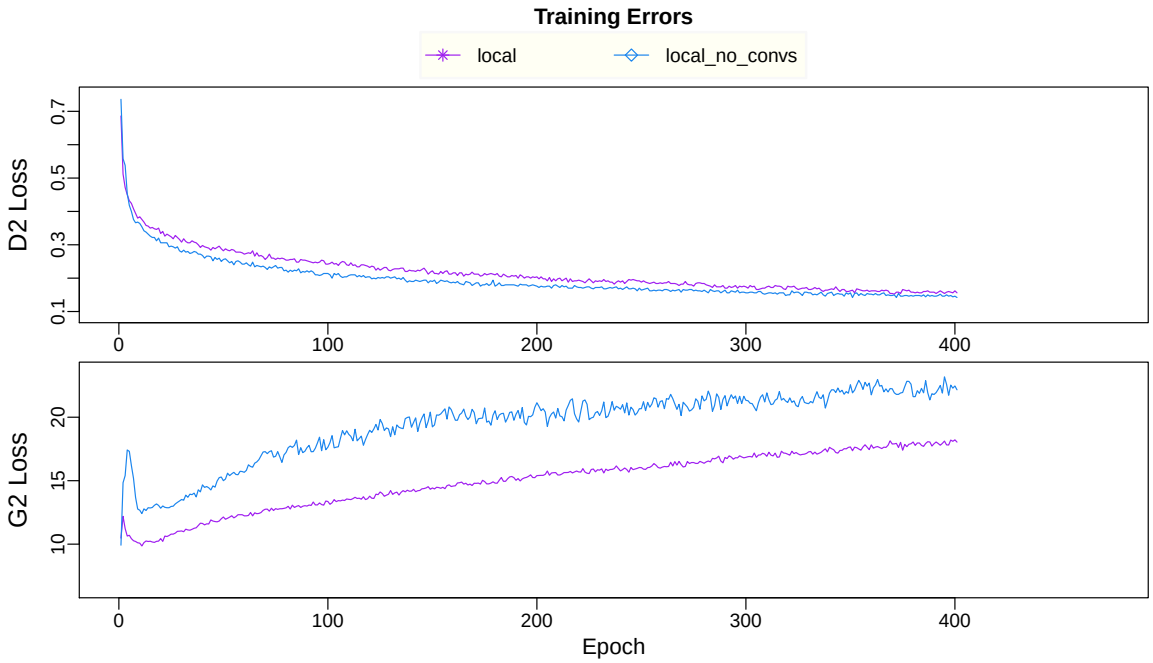

Figure 5.13.: Training losses of discriminator 2 and generator 2 (for 0 and 1 see Figure A.11) of replacing convolutions in the generators with local self-attention. [15]

[14] Figure was created by author.

5.3.7. Hyperparameter Tuning of our best Models

In the previous subsections two models showed promising results: local self-attention (see Subsection 5.3.1) and squeeze-and-excitation attention (see Subsection 5.3.3). All models in the previous subsections used the same set of hyperparameters optimised for the AttnGAN. For the new hyperparameters of the introduced attention models we followed their authors recommendations.

However, introducing other attention models and spectral normalisation changes the dynamic of the network. In addition, the context of the introduced attention models changed. Therefore, we tune our hyperparameters to achieve maximum performance.

The first hyperparameter we tune is λ. The authors of the AttnGAN [81] recommend a λ of 5.0. We decrease λ to 0.1.

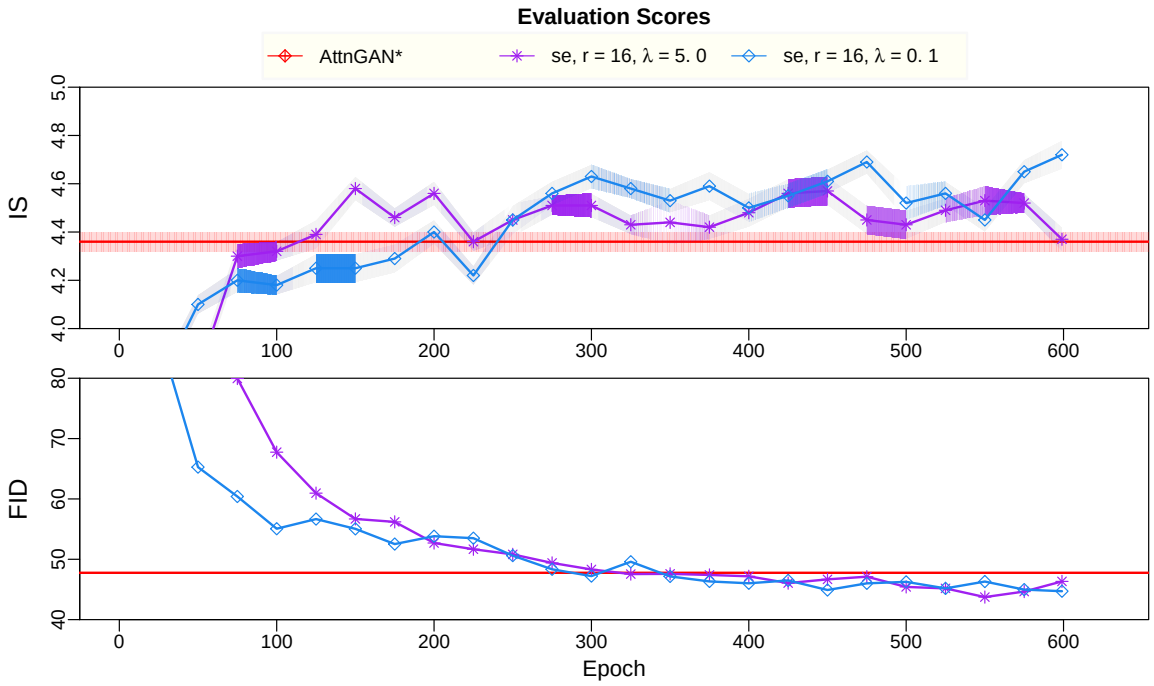

Figure 5.14.: IS and FID (see Figure A.12 for EMD, MMD, and 1-NN) for initial tuning of λ of our se attention model with an r of 16. [16]

For our se attention model we observe a positive impact on the 1-NN and MMD and a negative effect on the EMD (see Figure 5.14). Both the IS and FID yield comparable results, but the peak performance of both the best IS-FID (see Table A.1) and overall combination (see Table A.3) is better. We determine the best combination by maximising the sum of the relative improvements over the AttnGAN.

[15] Figure was created by author.
[16] Figure was created by author.

On the local self-attention model we observe positive impacts on the 1-NN, MMD, EMD, and FID, but a major negative impact on the IS (see Figure 5.18). However, we picked the local self-attention model for hyperparameter tuning due to its excellent performance on the IS. Since the performance of the IS (see Table A.2 for IS peak performances) is worse with a lower λ of 0.1 and the EMD and the FID are still not in a competitive region (see Table A.4), we prefer the λ of 5.0.

This tuning of the first hyperparameter demonstrates that different attention models react differently to hyperparameters indicating that they may ease or exacerbate learning in parts of the networks, for example in the DAMSM loss.

For our se attention model we experimented with tuning the internal hyperparameter r, which controls the reduction of the bottleneck layer in the se attention blocks. The authors[28] recommend an r of 16 that is ideally tuned for the actual network. We tested our network with an r of 16, 4, and 1, while having a λ of 0.1. With an r of 4 we observe slightly better results on the IS and similar results on the other evaluation metrics (see Figure 5.15). With an r of 1 we achieve even better results. Moreover, the peak performance of both the best IS-FID (see Table A.1) and overall combination (see Table A.3) is best at $r = 1$.

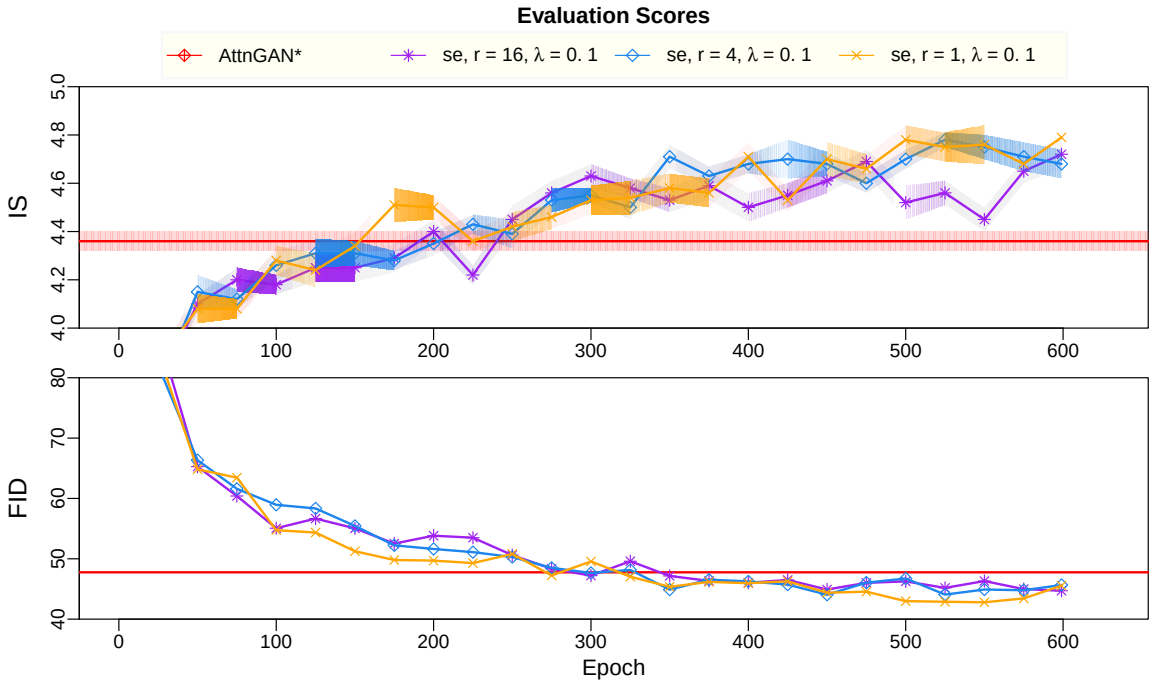

Figure 5.15.: IS and FID (see Figure A.13 for EMD, MMD, and 1-NN) for tuning the hyperparameter r, which controls the reduction of the bottleneck layer in the se attention blocks, of our se attention model with a λ of 0.1. [17]

[17] Figure was created by author.

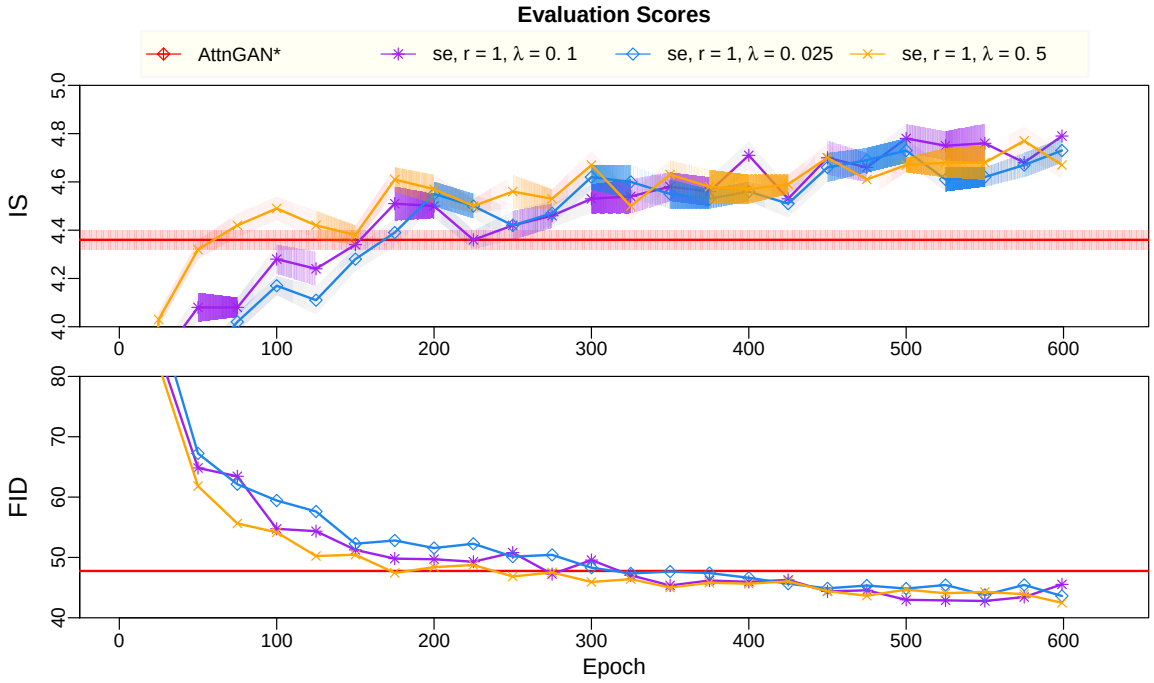

Figure 5.16.: IS and FID (see Figure A.14 for EMD, MMD, and 1-NN) for tuning of λ of our se attention model with an r of 1. [18]

After tuning r to 1, we experimented with tuning λ again. However, both an increase to 0.5 and a decrease to 0.025 yield similar results across all evaluation metrics (see Figure 5.16) and slightly worse peak performances of both the best IS-FID (see Table A.1) and overall combination (see Table A.3).

The authors of [83] suggest employing spectral normalisation not only in the discriminator, but also in the generator. However, we observe a worse response across all our evaluation metrics throughout training (see Figure 5.17) and on the peak performance (see Table A.1, Table A.3) for our se attention model.

Lastly, we combine local self-attention and se attention. Thereby, we only use local self-attention in the first generator and refrain from the use of se attention there. In the latter generators both attention models are applied as before: local self-attention is used in the attention module in conjunction with word attention and se attention is applied after every convolution except for convolutions used in attention mechanisms. We observe a positive impact on the IS and major negative impacts on the EMD and the FID. The average relative improvement over the AttnGAN turns negative for both the best IS-FID (see Table A.1) and overall combination (see Table A.3).

[18] Figure was created by author.

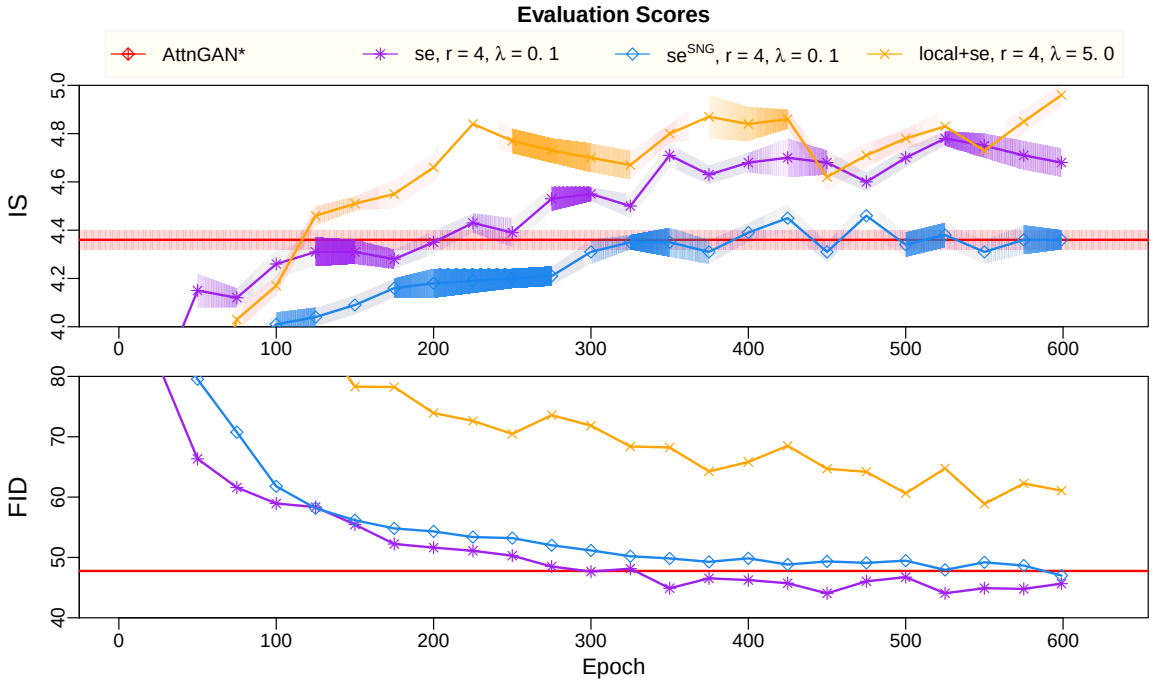

Figure 5.17.: IS and FID (see Figure A.15 for EMD, MMD, and 1-NN) for employing spectral normalisation in the generator (se^{SNG}) and for combining local-self attention and se attention. [19]

For our local self-attention model we focus on trying to boost the IS, because of the bad performance on the FID and EMD. As previously stated, decreasing λ to 0.1 had a negative impact on the IS.

In Subsection 5.3.4 we observe a minor positive impact on the IS using the height_max rather than the CBG method for our mixed model, which uses global self-attention in the early stages and local self-attention in the later stages. For our local self-attention model, we observe slightly worse scores on the IS and similar scores on our evaluation metrics when using the height_max method. Furthermore, the peak performance of the IS is worse (see Table A.2).

Finally, we combine local self-attention and se attention (see above). We observe a positive impact on all evaluation metrics. While the EMD and FID are not competitive, the IS is boosted from 4.81 to 4.96.

[19] Figure was created by author.

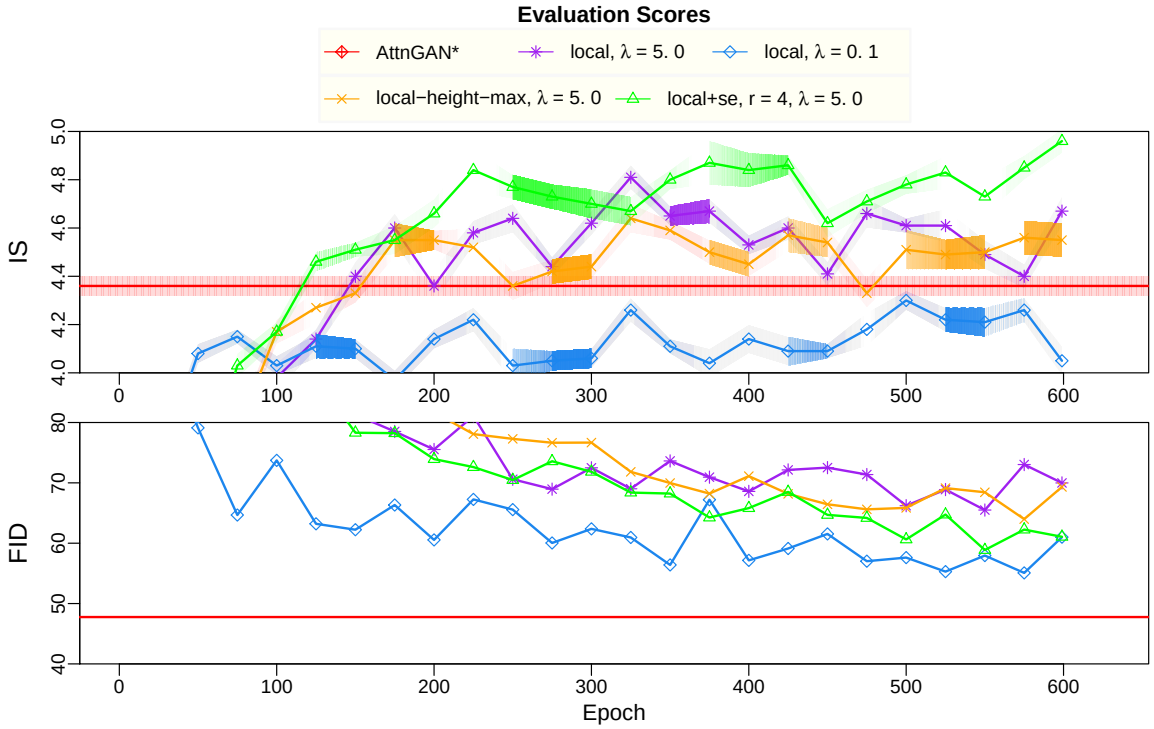

Figure 5.18.: IS and FID (see Figure A.16 for EMD, MMD, and 1-NN) of various local self-attention models with different hyperparameters and of local self-attention with se attention. [20]

In conclusion, judging by the best IS-FID and overall combination our best model is se attention with $r = 1, \lambda = 0.1$ yielding significant average relative improvements of 9.8% and 6.8%, respectively, over the AttnGAN. Combining local self-attention and se attention achieves the best IS of 4.96 at the cost of significant negative improvements on the FID and EMD.

[20] Figure was created by author.

5.3.8. Visual Analysis of our best Models

Figure 5.19.: Examples of images generated by (a) AttnGAN, (b) our se attention model with $r = 1, \lambda = 0.1$, (c) our local self-attention model with $\lambda = 5.0$, and (d) our combined model of se attention and local self-attention with $r = 4, \lambda = 5.0$ conditioned on text descriptions from the CUB test set and (e) the corresponding ground truth. [21]

In the previous sections we solely relied on quantitative metrics to evaluate our models. Here, we perform qualitative tests of our best models. Figure 5.19 presents a subjective visual comparison among the AttnGAN, our se attention model with $r = 1, \lambda = 0.1$, our local self-attention model with $\lambda = 5.0$, our combined model of se attention and local self-attention with $r = 4, \lambda = 5.0$, and the corresponding ground truth.

Figure 5.19 shows that images generated by the AttnGAN are of great detail (4^{th}, 5^{th}, 6^{th}, and 7^{th} column), but are also cut off (1^{st} and 2^{nd} column), colors are inconsistent with the text descriptions (7^{th} and 8^{th} column), birds melt with their surrounding (2^{nd} column), and birds are drawn strangely (3^{rd} and 4^{th} column). Our se attention model generates images of similar quality with similar mistakes (see 3^{rd}, 5^{th}, and 7^{th} column for cut-offs, 7^{th} and 8^{th} column for color inconsistencies, etc.) while scoring better across all evaluation metrics except the EMD. Figure 5.19 also shows that the test data is not perfect either: images are cut off as well (1^{st}, 3^{rd}, and 4^{th} column) and the blue head of the bird is missing

[21] Figure was created by author. AttnGAN images generated using the official model.

in the corresponding text description (2^{nd} column).

Both models incorporating local self-attention fail to produce realistic looking image, despite scoring higher ISs than the AttnGAN and our se attention model. Instead, they draw repetitive features manifesting in the form of multiple birds, drawn out birds, multiple heads, or strange patterns. The drawn features mostly match the textual descriptions. This provides a possible explanation why both models have a high IS despite scoring poorly on the other evaluation metrics: the IS cares mainly about the images being highly classifiable and diverse. Thereby, it presumes that highly classifiable images are of high quality. Our networks demonstrate that high classify-ability and diversity and therefore a high IS can be achieved through completely unrealistic, repetitive features of the correct bird class. This is further evidence that improvements solely based on the IS have to be viewed sceptically.

For our se attention model we further test its generalisation ability by testing how sensitive the outputs are to changes in the most attended, in the sense of word attention, words in the text descriptions (see Figure 5.20). The test is similar to the one performed on the AttnGAN [81]. The results illustrate that adding se attention and spectral normalisation do not harm the generalisation ability of the network: the images are altered according to the changes in the input sentences, showing that the network retains its ability to react to subtle semantic differences in the text descriptions.

Additional images of our models during training and at their peak performance are Figure A.17, Figure A.18, Figure A.19.

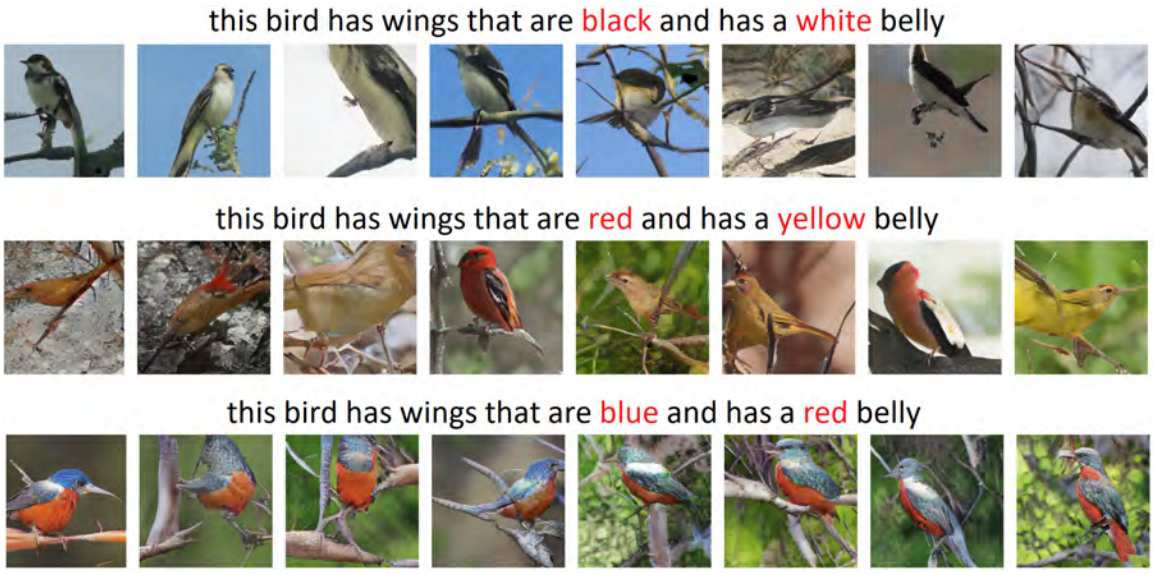

Figure 5.20.: Example results of our se attention model with $r = 1, \lambda = 0.1$ trained on the CUB dataset while changing some most attended, in the sense of word attention, words in the text descriptions. [22]

5.4. Comparison to the state of the art

Table 5.4.: Fréchet Inception Distance (FID) and Inception Score (IS) of state-of-the-art models and our two CAGAN models on the CUB dataset with a 256x256 image resolution.

Model	IS↑	FID↓
Real Data	25.52 ± 0.09	0.00
GAWWN [56]	3.62 ± 0.07	67.22
StackGAN-v1 [84]	3.70 ± 0.04	51.89
StackGAN-v2 [85]	3.82 ± 0.06	–
AttnGAN [81]	4.36 ± 0.04	47.76[23]
PPAN [39]	4.38 ± 0.05	–
HAGAN [10]	4.43 ± 0.03	44.64[24]
MirrorGAN [53]	4.56 ± 0.05	–
ControlGAN [38]	4.58 ± 0.09	–
DualAttn-GAN [8]	4.59 ± 0.07	14.06[25]
LeicaGAN [52]	4.62 ± 0.06	–
SD-GAN [82]	4.67 ± 0.09	–
DM-GAN [86]	4.75 ± 0.07	16.09[26]
CAGAN_SE (ours)	4.78 ± 0.06	**42.98**
CAGAN_L+SE (ours)	**4.96 ± 0.05**	61.06

Table 5.4 compares our two best models squeeze-and-excitation attention and squeeze-and-excitation attention combined with local self-attention to the state-of-the-art models. Our squeeze-and-excitation attention model boosts the IS of our baseline by 9.6% ± 2.4% from 4.36 ± 0.04 to 4.78 ± 0.06 and improves the state of the art by 0.6% ± 2.8% from 4.75 ± 0.07 to 4.78 ± 0.06; and it boosts the FID of our baseline by 10.0% from 47.76 to 42.98. A comparison to the FIDs of the state of the art is futile, because several papers report no FID score and those that do report vastly different FID scores on the CUB dataset for the same baseline suggesting the use of different FID implementations (see Table 5.5).

Our combined model boosts the IS of our baseline by 13.8% ± 2.2% from 4.36 ± 0.04 to 4.96 ± 0.05 and improves the state of the art by 4.4% ± 2.6% from 4.75 ± 0.07 to 4.96 ± 0.05. However, it generates completely unrealistic images through feature repetitions (see Subsection 5.3.8) and has a major negative impact on the FID of our baseline of 27.8% from 47.76 to 61.06. This demonstrates the importance of reporting both scores.

[22] Figure was created by author.
[23] Not an officially reported score. Re-evaluated using the official model.
[24] Reported a slightly different baseline FID of the AttnGAN (see Table 5.5).
[25] Reported a different baseline FID of the AttnGAN (see Table 5.5).
[26] Reported a different baseline FID of the AttnGAN (see Table 5.5).

Table 5.5.: Fréchet Inception Distance (FID) of the AttnGAN on the CUB dataset with a 256x256 image resolution reported by respective papers. The AttnGAN paper itself does not report an FID score.

Paper/Model	FID of the AttnGAN
AttnGAN [81]	–
HAGAN [10]	46.43
DualAttn-GAN [8]	16.48
DM-GAN [86]	23.98
CAGAN (ours)	47.76

Table 5.5 lists the Fréchet Inception Distance of the AttnGAN on the CUB dataset reported by recent state-of-the-art papers. We observe that the three papers report vastly different FID scores for the same network, on the same dataset, with the same split, with the same image resolution, and with an almost identical number of roughly 30k samples. With such a different baseline, any comparison of the reported FID scores of the respective papers is futile.

Our measurement of the AttnGAN's FID is closest to the HAGAN measurement [10]. The difference of nearly 3% may result from different seeds or the use of the tensorflow implementation of the Inception v3 network instead of the pytorch implementation.

6. Conclusion

We proposed combining multiple attention models in the context of text-to-image generation with stacked Generative Adversarial Networks (GANs). These models included global, local, and light-weight self-attention on feature maps as well as linear and grid attention repurposed for sentence attention.

We evaluated our proposal using several of the most popular evaluation metrics for generative image modelling, including the Inception Score (IS) and the Fréchet Inception Distance (FID). By combining squeeze-and-excitation attention with word attention and applying spectral normalisation, a GAN stabilising technique, our proposed Combined Attention Generative Adversarial Network (CAGAN) boosted the IS of our baseline (the AttnGAN) by 9.6% ± 2.4% from 4.36 ± 0.04 to 4.78 ± 0.06 and improved the state of the art by 0.6% ± 2.8% from 4.75 ± 0.07 to 4.78 ± 0.06 on the CUB dataset.

Furthermore, our proposed CAGAN boosted the FID of our baseline by 10.0% from 47.76 to 42.98. A comparison to the FIDs of the state of the art is futile, because several papers report no FID score and those that do report vastly different FID scores on the CUB dataset for the same baseline suggesting the use of different FID implementations.

We demonstrated that these alterations change the training behaviour of the network, such as increasing the learn-ability of certain parts of the loss function. We showed that these altered networks benefit from a new set of optimised hyperparameters. Future work may include further hyperparameter tuning and a better understanding on the impact of attention models on the individual parts of the loss function.

We critically discussed several evaluation metrics for text-to-image generation and analysed their anti-correlation by searching for opposing responses, i.e., occurrences of improving on one metric while deteriorating on another metric. Our findings demonstrate that evaluation metrics may vary significantly and that relative improvements on specific metrics, especially relative improvements, have to be viewed sceptically. Moreover, we managed to create a model boosting our baseline on one specific evaluation metric, the IS, by 13.8% ± 2.2% from 4.36 ± 0.04 to 4.96 ± 0.05 while generating completely unrealistic images through feature repetitions and having a major negative impact on the FID of our baseline of 27.8% from 47.76 to 61.06.

We observed internal anti-correlation in our experiments and demonstrated that the choice of the evaluation metric or even the choice of the combination of evaluation metrics may lead to different model judgements. Our findings emphasize the need for the use of more than one evaluation metric; a unified evaluation approach in the field of text-to-image

generation; and ideally an evaluation metric offering a fair model comparison.

Future work may also include gaining a deeper understanding of different and coherent behaviour of evaluation metrics; how attention models impact different parts of networks; and how different parts of networks influence evaluation metrics.

In our experiments using attention in the discriminator led to mode collapse. Investigating this behaviour and developing attention models for the discriminator remains future work. Replacing convolutions with local self-attention in the generators also showed no learning across the evaluation metrics. However, a preliminary analysis revealed that instead of a mode collapse the network does slowly learn its loss functions. A deeper analysis may explain why the evaluation metrics are not reflecting this behaviour and its cause.

Future work may also include generating 3D data which is then rendered to an image instead of directly generating the image. This results in a more meaningful yet complex representation, may reduce artefacts, and may generate images of higher quality.

A. Appendix

A.1. Experimental Results and Evaluation

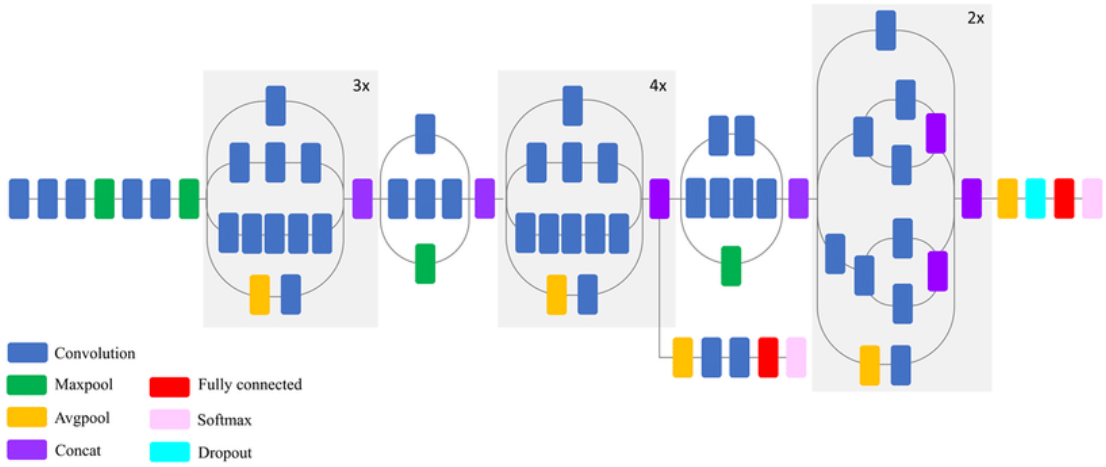

Figure A.1.: Schematic diagram of the Inception v3 network. [1]

[1]Figure taken from [43].

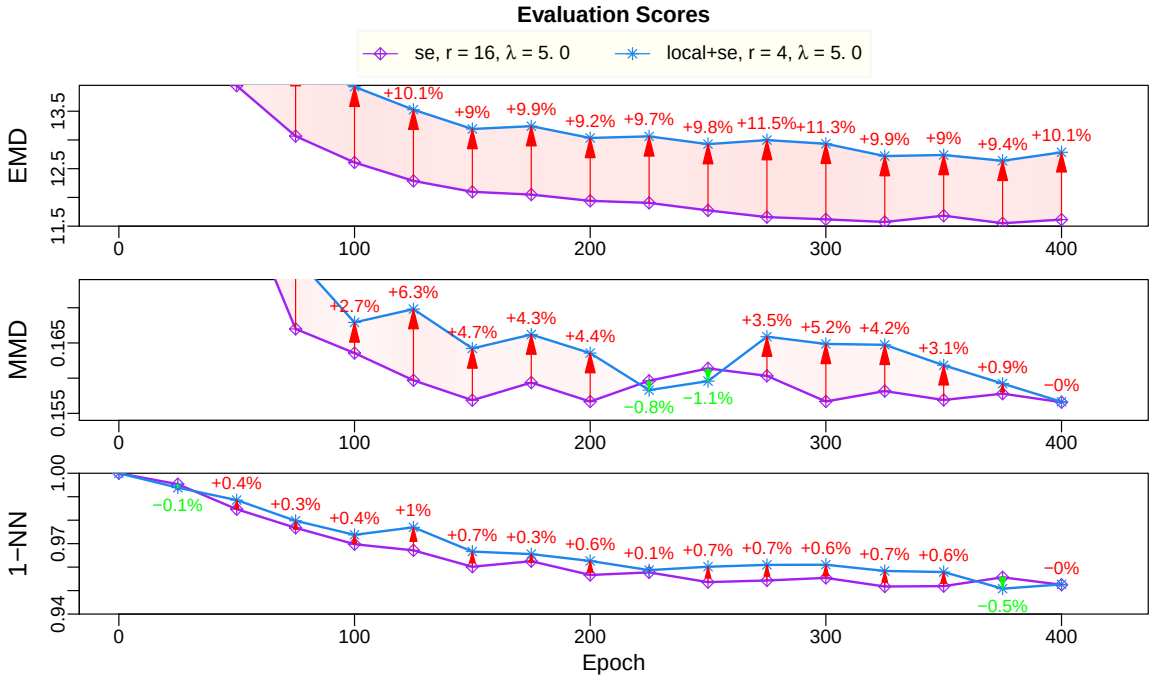

Figure A.2.: EMD, MMD, and NN-1 for Figure 5.3. [2]

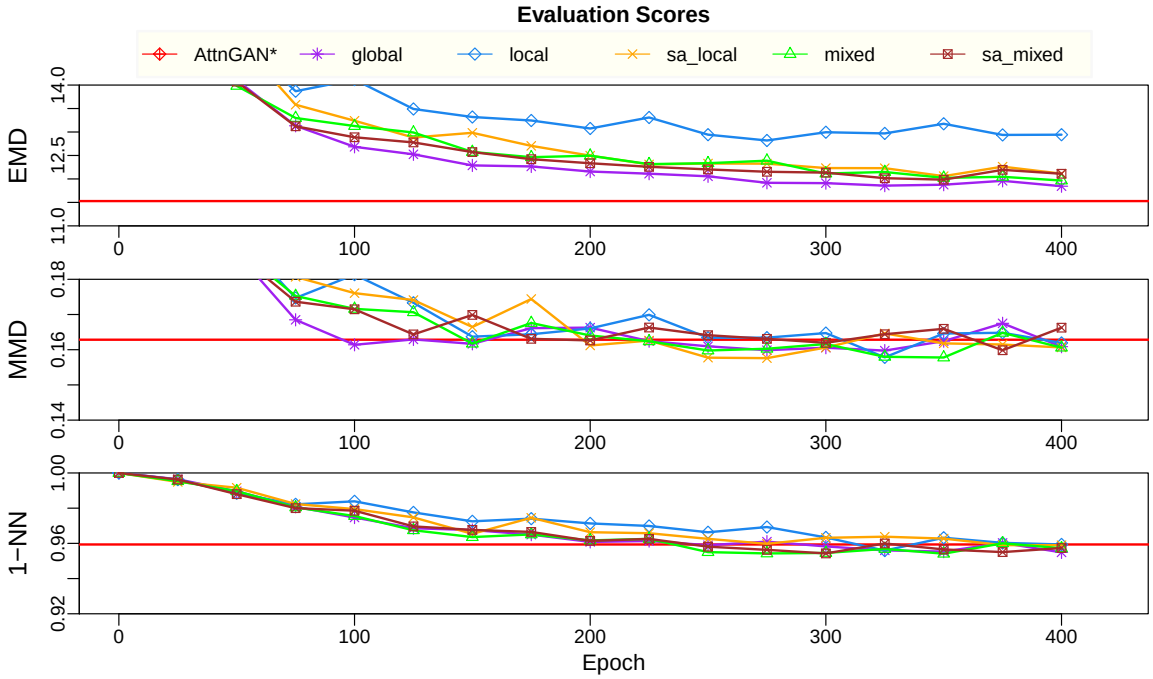

Figure A.3.: EMD, MMD, and NN-1 for Figure 5.5. [3]

[2]Figure was created by author.

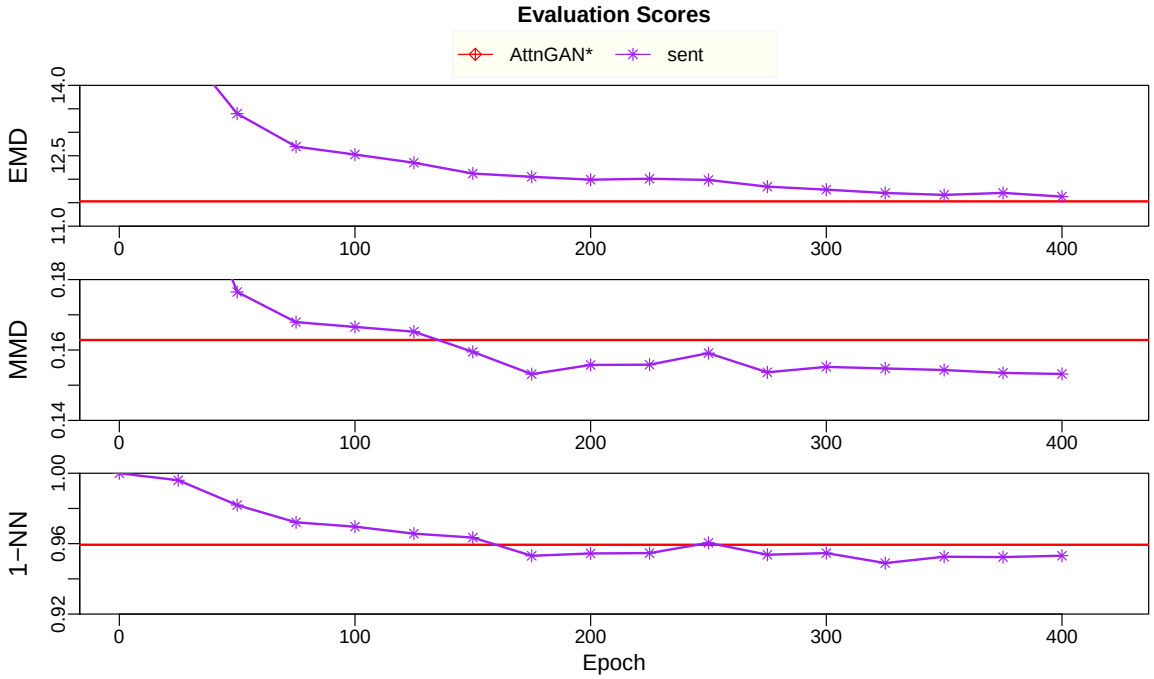

Figure A.4.: EMD, MMD, and NN-1 for Figure 5.6. [4]

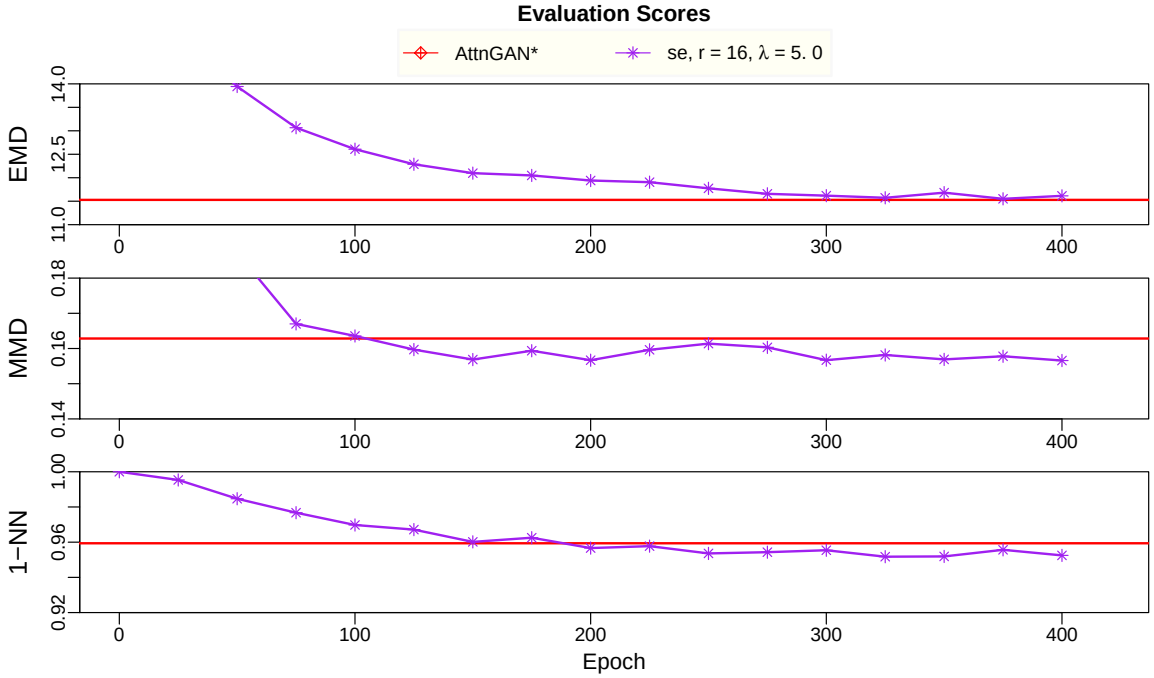

Figure A.5.: EMD, MMD, and NN-1 for Figure 5.7. [5]

[3]Figure was created by author.
[4]Figure was created by author.

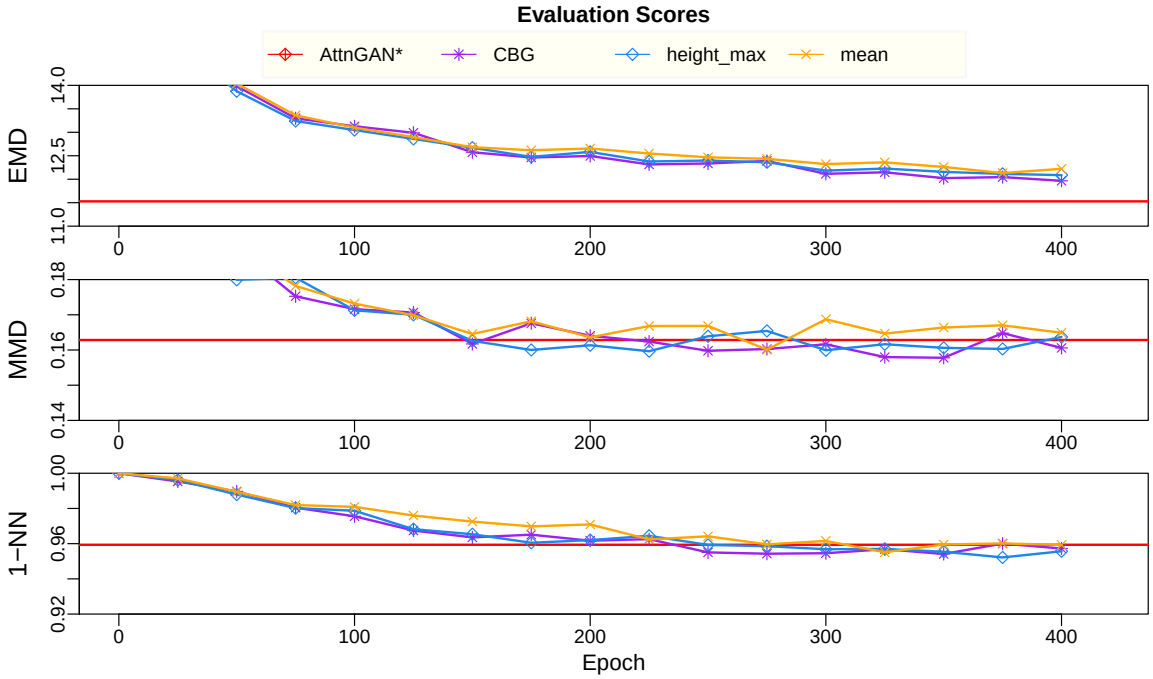

Figure A.6.: EMD, MMD, and NN-1 for Figure 5.8. [6]

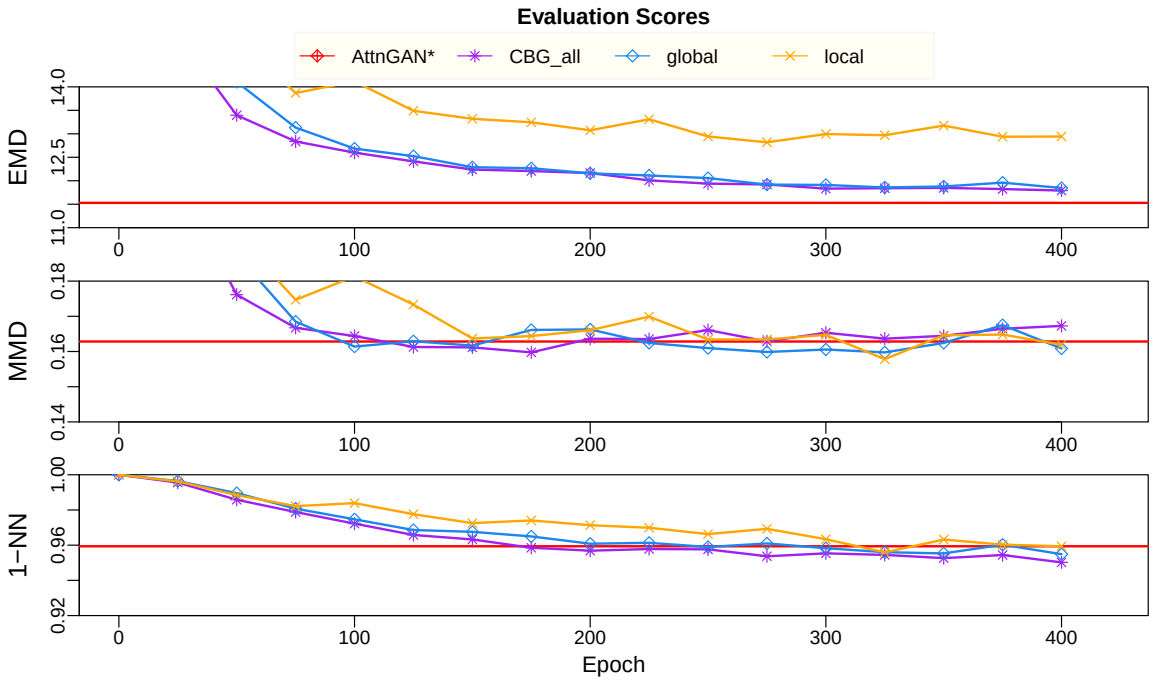

Figure A.7.: EMD, MMD, and NN-1 for Figure 5.9. [7]

[5]Figure was created by author.
[6]Figure was created by author.

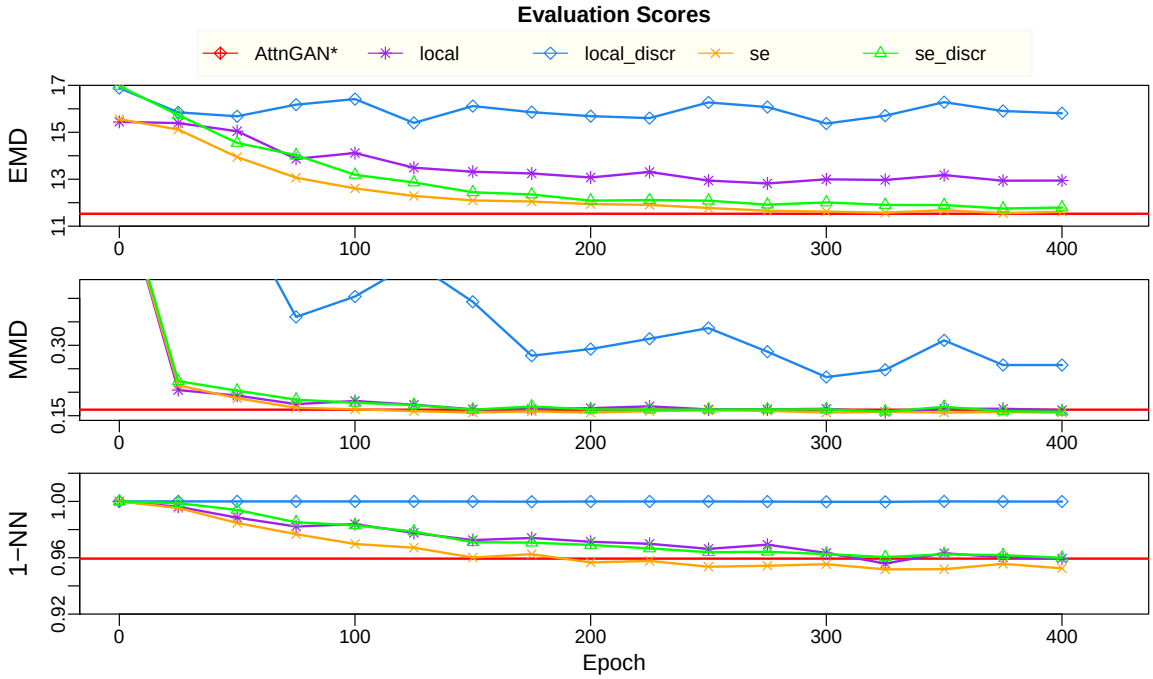

Figure A.8.: EMD, MMD, and NN-1 for Figure 5.10. [8]

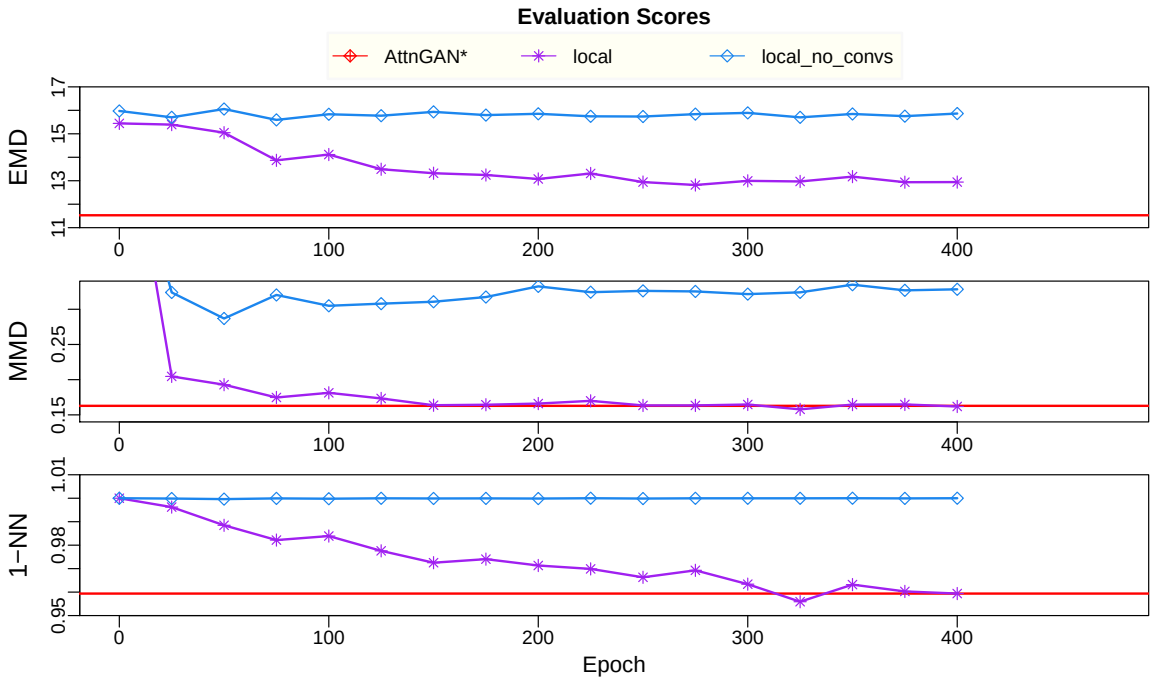

Figure A.9.: EMD, MMD, and NN-1 for Figure 5.12. [9]

[7] Figure was created by author.
[8] Figure was created by author.

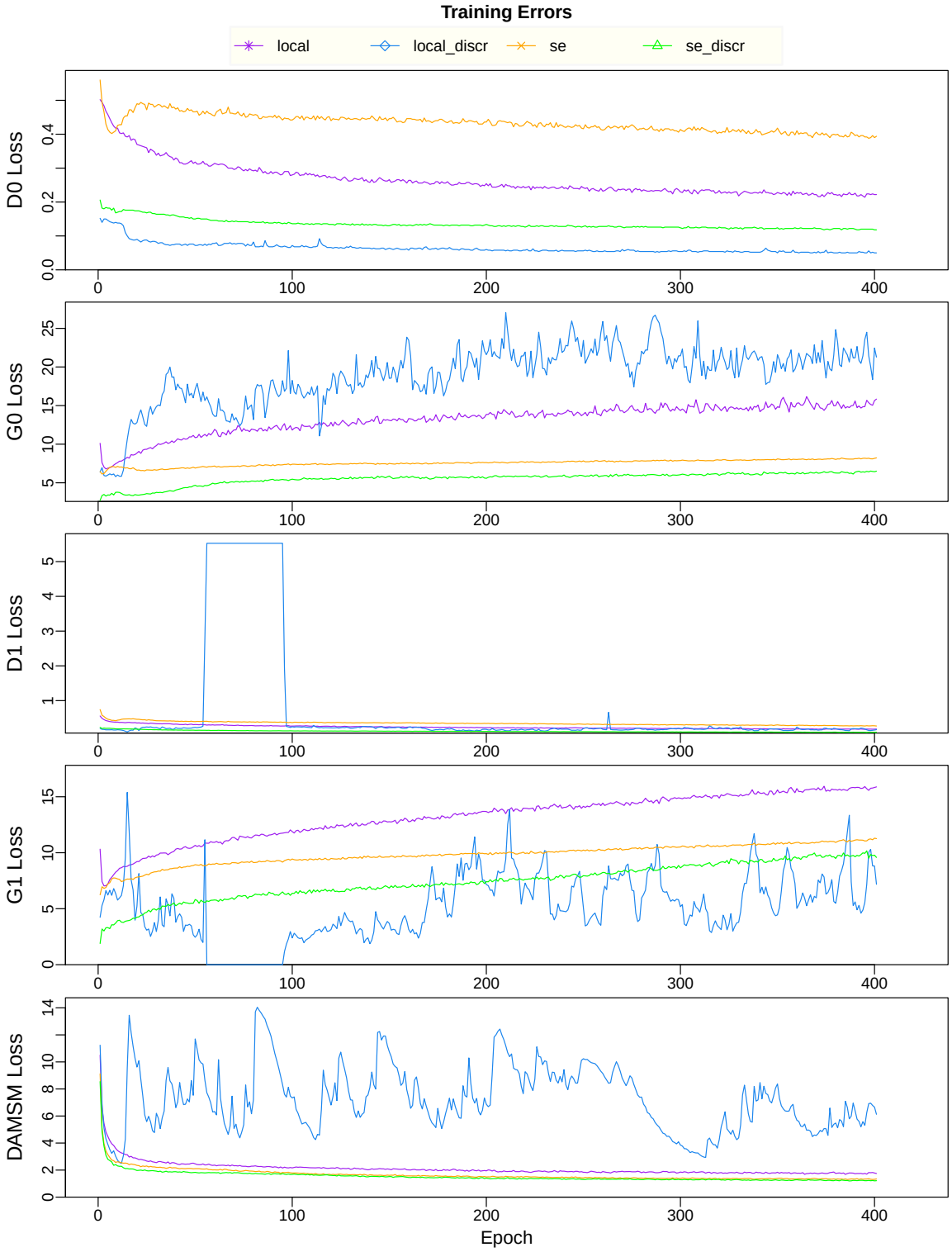

Figure A.10.: Training losses of discriminators 0 and 1, generators 0 and 1, and DAMSM loss for Figure 5.11. [10]

[9]Figure was created by author.

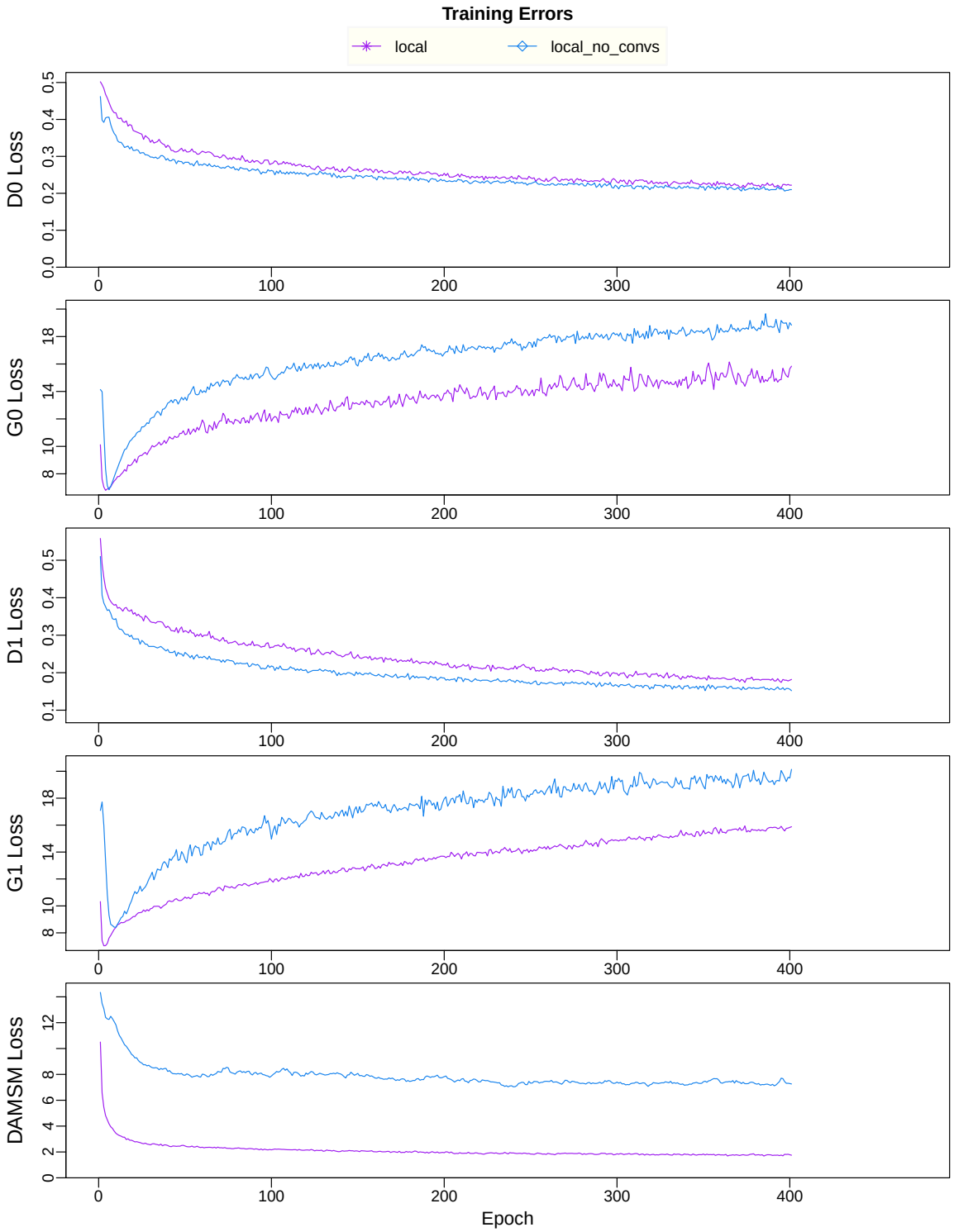

Figure A.11.: Training losses of discriminators 0 and 1, generators 0 and 1, and DAMSM loss for Figure 5.13. [11]

[10]Figure was created by author.

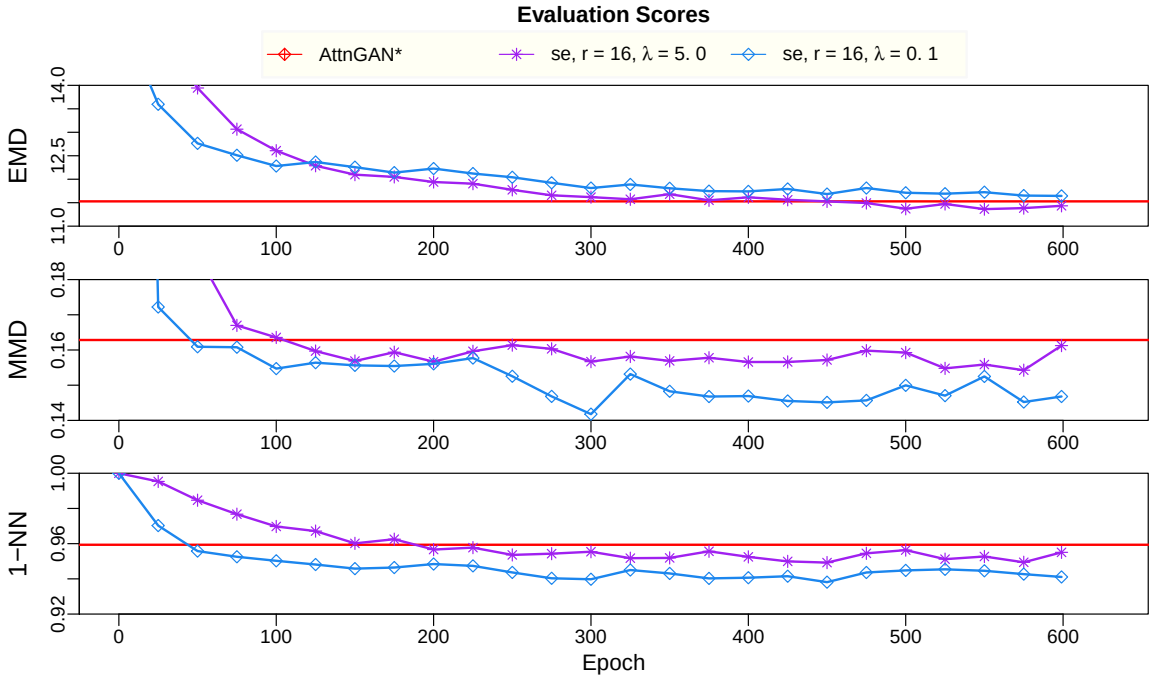

Figure A.12.: EMD, MMD, and NN-1 for Figure 5.14. [12]

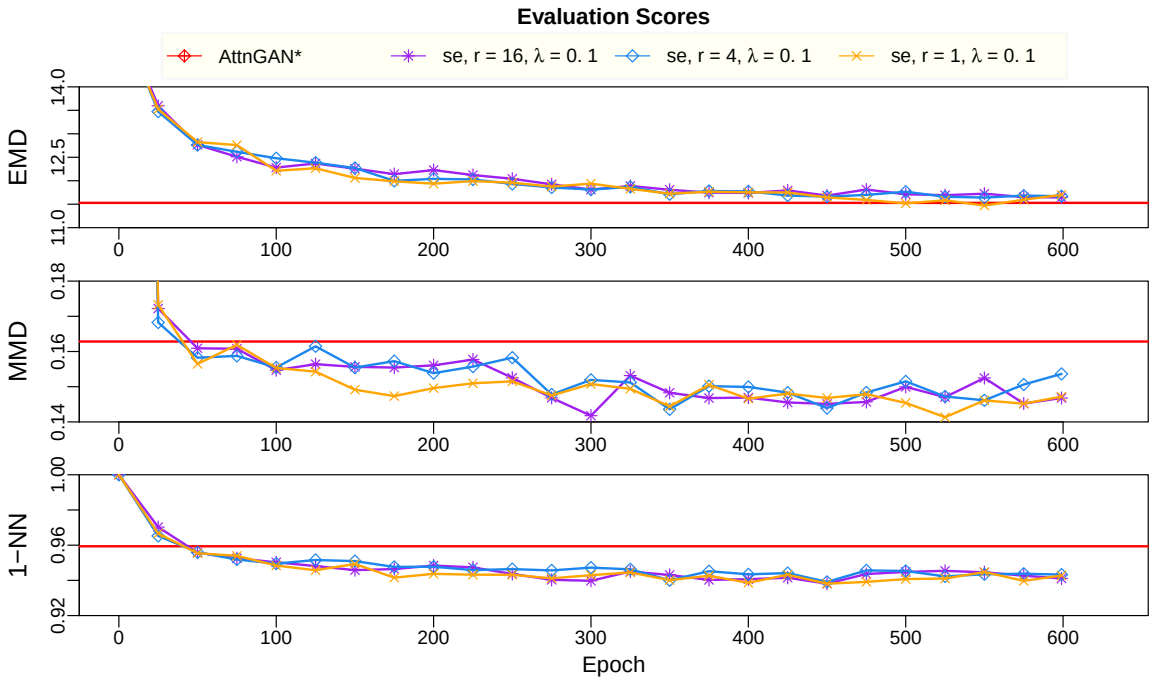

Figure A.13.: EMD, MMD, and NN-1 for Figure 5.15. [13]

[11] Figure was created by author.
[12] Figure was created by author.

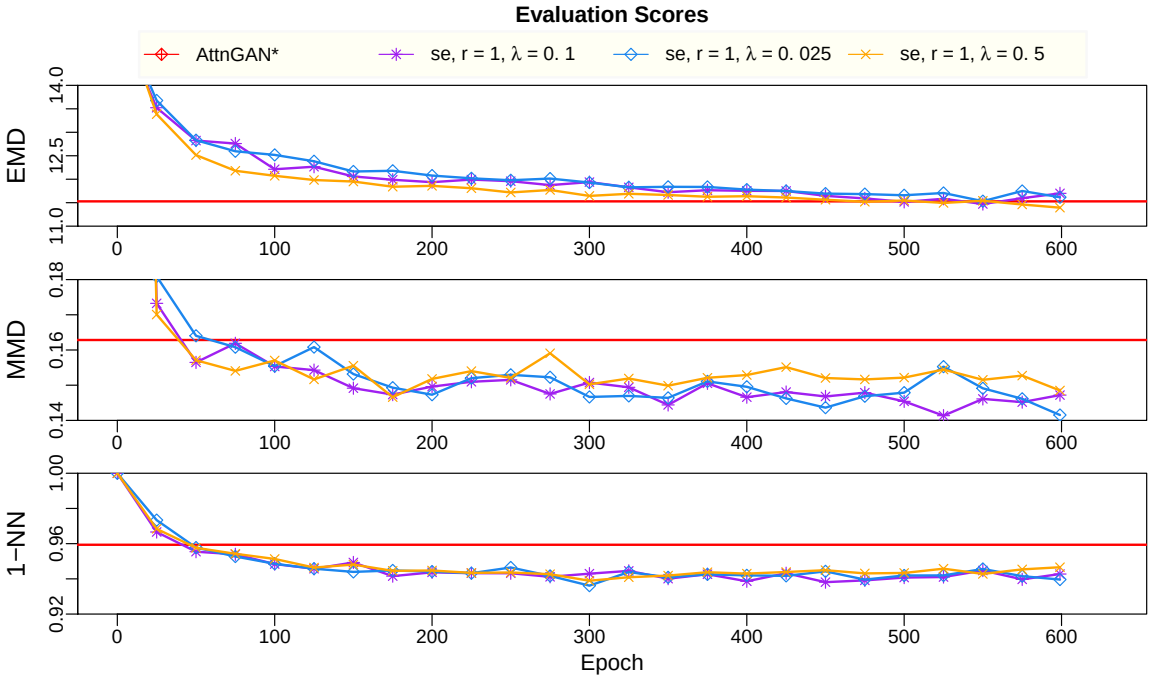

Figure A.14.: EMD, MMD, and NN-1 for Figure 5.16. [14]

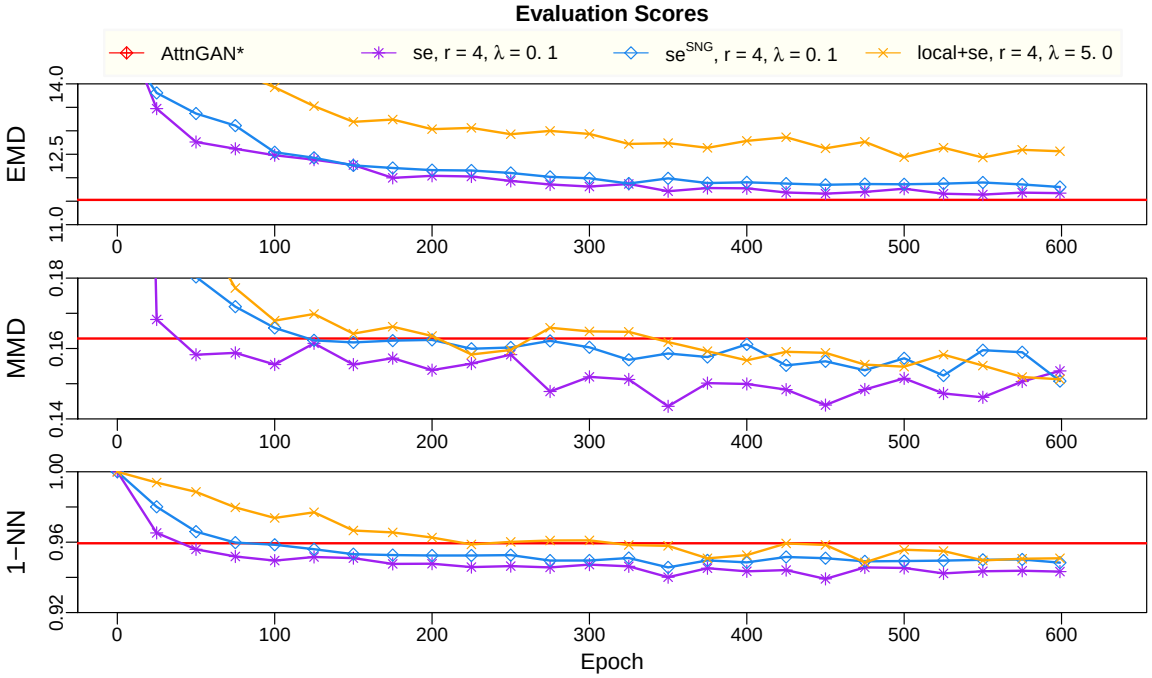

Figure A.15.: EMD, MMD, and NN-1 for Figure 5.17. [15]

[13] Figure was created by author.
[14] Figure was created by author.

Figure A.16.: EMD, MMD, and NN-1 for Figure 5.18. [16]

[15] Figure was created by author.
[16] Figure was created by author.

Table A.1.: Best IS-FID combination of our se attention models and their relative improvements over the AttnGAN.

Model	Epoch		IS↑	FID↓	EMD↓	MMD↓	1-NN↓↑	⌀
se, $r = 16$, $\lambda = 5.0$	550	Score	4.53	43.73	11.36	0.156	0.953	-
		r	+3.9%	+8.4%	+1.5%	+4.2%	+0.7%	+6.2%
se, $r = 16$, $\lambda = 0.1$	599	Score	4.72	44.71	11.64	0.147	0.941	-
		r	+8.3%	+6.4%	−1.0%	+9.7%	+1.9%	+7.3%
se, $r = 4$, $\lambda = 0.1$	525	Score	4.78	44.07	11.66	0.147	0.942	-
		r	+9.6%	+7.7%	−1.1%	+9.7%	+1.8%	+8.7%
se, $r = 1$, $\lambda = 0.1$	500	Score	4.78	42.98	11.52	0.145	0.941	-
		r	+9.6%	+10.0%	+0.1%	+11.0%	+1.9%	+9.8%
se, $r = 1$, $\lambda = 0.025$	599	Score	4.73	43.61	11.62	0.142	0.940	-
		r	+8.5%	+8.7%	−0.8%	+12.8%	+2.0%	+8.6%
se, $r = 1$, $\lambda = 0.5$	599	Score	4.67	42.49	11.39	0.148	0.947	-
		r	+7.1%	+11.0%	+1.2%	+9.1%	+1.3%	+9.1%
se^{SNG}, $r = 4$, $\lambda = 0.1$	599	Score	4.36	47.00	11.80	0.151	0.948	-
		r	+0.0%	+1.6%	−2.4%	+7.3%	+1.2%	+0.8%
local+se, $r = 4$, $\lambda = 5.0$	599	Score	4.96	61.06	12.56	0.151	0.951	-
		r	+13.8%	−27.8%	−8.9%	+7.3%	+0.9%	−7.0%

Table A.2.: Best Inception Scores of our local self-attention models and their relative improvements over the AttnGAN.

Model	Epoch		IS↑	FID↓	EMD↓	MMD↓	1-NN↓↑	⌀
local, $\lambda = 5.0$	325	Score	4.81	69.01	12.97	0.158	0.956	-
		r	+10.3%	−44.5%	−12.5%	+3.0%	+0.4%	+10.3%
local, $\lambda = 0.1$	225	Score	4.22	67.26	12.76	0.156	0.950	-
		r	−3.2%	−40.8%	−10.7%	+4.2%	+1.0%	−3.2%
local_height_max, $\lambda = 5.0$	325	Score	4.64	71.83	13.02	0.157	0.962	-
		r	+6.4%	−50.4%	−12.9%	+3.6%	−0.3%	+6.4%
local+se, $r = 4$, $\lambda = 5.0$	599	Score	4.96	61.06	12.56	0.151	0.951	-
		r	+13.8%	−27.8%	−8.9%	+7.3%	+0.9%	+13.8%

Table A.3.: Best overall combination of our se attention models and their relative improvements over the AttnGAN.

Model	Epoch		IS↑	FID↓	EMD↓	MMD↓	1-NN↓	⌀
se, $r = 16$, $\lambda = 5.0$	550	Score	4.53	43.73	11.36	0.156	0.953	-
		r	+3.9%	+8.4%	+1.5%	+4.2%	+0.7%	+3.7%
se, $r = 16$, $\lambda = 0.1$	599	Score	4.72	44.71	11.64	0.147	0.941	-
		r	+8.3%	+6.4%	−1.0%	+9.7%	+1.9%	+5.1%
se, $r = 4$, $\lambda = 0.1$	450	Score	4.68	44.05	11.66	0.144	0.939	-
		r	+7.3%	+7.8%	−1.1%	+11.6%	+2.1%	+5.5%
se, $r = 1$, $\lambda = 0.1$	525	Score	4.75	42.88	11.58	0.141	0.941	-
		r	+8.9%	+10.2%	−0.4%	+13.4%	+1.9%	+6.8%
se, $r = 1$, $\lambda = 0.025$	599	Score	4.73	43.61	11.62	0.142	0.940	-
		r	+8.5%	+8.7%	−0.8%	+12.8%	+2.0%	+6.3%
se, $r = 1$, $\lambda = 0.5$	599	Score	4.67	42.49	11.39	0.148	0.947	-
		r	+7.1%	+11.0%	+1.2%	+9.1%	+1.3%	+5.9%
se^{SNG}, $r = 4$, $\lambda = 0.1$	599	Score	4.36	47.00	11.80	0.151	0.948	-
		r	+0.0%	+1.6%	−2.4%	+7.3%	+1.2%	+1.6%
local+se, $r = 4$, $\lambda = 5.0$	599	Score	4.96	61.06	12.56	0.151	0.951	-
		r	+13.8%	−27.8%	−8.9%	+7.3%	+0.9%	−3.0%

Table A.4.: Best overall combination of our local self-attention models and their relative improvements over the AttnGAN.

Model	Epoch		IS↑	FID↓	EMD↓	MMD↓	1-NN↓	⌀
local, $\lambda = 5.0$	500	Score	4.61	66.22	12.79	0.160	0.959	-
		r	+5.7%	−38.6%	−10.9%	+1.7%	+0.0%	−8.5%
local, $\lambda = 0.1$	275	Score	4.05	60.04	12.50	0.151	0.949	-
		r	−7.1%	−25.7%	−8.4%	+7.3%	+1.1%	−6.6%
local_height_max, $\lambda = 5.0$	575	Score	4.56	63.99	12.71	0.161	0.955	-
		r	+4.6%	−34.0%	−10.2%	+1.1%	+0.5%	−7.7%
local+se, $r = 4$, $\lambda = 5.0$	599	Score	4.96	61.06	12.56	0.151	0.951	-
		r	+13.8%	−27.8%	−8.9%	+7.3%	+0.9%	−3.0%

Figure A.17.: 16 images generated from random captions of the test dataset for epochs 250 (left) and 500 (right) of our se attention model, $r = 1, \lambda = 0.1$. [17]

Figure A.18.: 16 images generated from random captions of the test dataset for epochs 175 (left) and 325 (right) of our local self-attention model, $\lambda = 5.0$. [18]

[17] Figure was created by author.
[18] Figure was created by author.

Figure A.19.: 16 images generated from random captions of the test dataset for epochs 300 (left) and 599 (right) of our se attention combined with local self-attention model, $r = 4, \lambda = 5.0$. [19]

[A][Da][Pr][S][W][pres][I]

[19] Figure was created by author.

Bibliography

[1] Martín Arjovsky and Léon Bottou. "Towards Principled Methods for Training Generative Adversarial Networks". In: *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, Conference Track Proceedings*. 2017.

[2] Martín Arjovsky, Soumith Chintala, and Léon Bottou. "Wasserstein Generative Adversarial Networks". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, August 6-11*. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 214–223.

[3] Shuang Bai and Shan An. "A survey on automatic image caption generation". In: *Neurocomputing* 311 (2018), pp. 291–304.

[4] Shane T. Barratt and Rishi Sharma. "A Note on the Inception Score". In: *CoRR* abs/1801.01973 (2018).

[5] Yoshua Bengio, Pascal Lamblin, Dan Popovici, and Hugo Larochelle. "Greedy Layer-Wise Training of Deep Networks". In: *Advances in Neural Information Processing Systems 19, Proceedings of the Twentieth Annual Conference on Neural Information Processing Systems, Vancouver, BC, Canada, December 4-7*. 2006, pp. 153–160.

[6] John Blitzer, Ryan T. McDonald, and Fernando Pereira. "Domain Adaptation with Structural Correspondence Learning". In: *EMNLP 2006, Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing, Sydney, Australia, July 22-23*. 2006, pp. 120–128.

[7] Peter F. Brown, Vincent J. Della Pietra, Peter V. de Souza, Jennifer C. Lai, and Robert L. Mercer. "Class-Based n-gram Models of Natural Language". In: *Computational Linguistics* 18.4 (1992), pp. 467–479.

[8] Yali Cai, Xiaoru Wang, Zhihong Yu, Fu Li, Peirong Xu, Yueli Li, and Lixian Li. "Dualattn-GAN: Text to Image Synthesis With Dual Attentional Generative Adversarial Network". In: *IEEE Access* 7 (2019), pp. 183706–183716.

[9] Qifeng Chen and Vladlen Koltun. "Photographic Image Synthesis with Cascaded Refinement Networks". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29*. 2017, pp. 1520–1529.

[10] Qingrong Cheng and Xiaodong Gu. "Hybrid Attention Driven Text-to-Image Synthesis via Generative Adversarial Networks". In: *Artificial Neural Networks and Machine Learning - ICANN 2019 - 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17-19, Proceedings - Workshop and Special Sessions*. Vol. 11731. Lecture Notes in Computer Science. Springer, 2019, pp. 483–495.

[11] Yunjey Choi, Min-Je Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. "StarGAN: Unified Generative Adversarial Networks for Multi-Domain Image-to-Image Translation". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22*. 2018, pp. 8789–8797.

[12] Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. "Language Modeling with Gated Convolutional Networks". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, August 6-11*. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 933–941.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), Miami, Florida, USA, June 20-25*. IEEE Computer Society, 2009, pp. 248–255.

[14] Emily L. Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. "Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montréal, QC, Canada, December 7-12*. 2015, pp. 1486–1494.

[15] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *CoRR* abs/1810.04805 (2018).

[16] Garoe Dorta, Sara Vicente, Lourdes Agapito, Neill D. F. Campbell, Simon Prince, and Ivor Simpson. "Laplacian Pyramid of Conditional Variational Autoencoders". In: *Proceedings of the 14th European Conference on Visual Media Production (CVMP 2017), London, United Kingdom, December 11-13*. 2017, 7:1–7:9.

[17] Hao Fang, Saurabh Gupta, Forrest N. Iandola, Rupesh Kumar Srivastava, Li Deng, Piotr Dollár, Jianfeng Gao, Xiaodong He, Margaret Mitchell, John C. Platt, C. Lawrence Zitnick, and Geoffrey Zweig. "From captions to visual concepts and back". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12*. 2015, pp. 1473–1482.

[18] Ian J. Goodfellow. "NIPS 2016 Tutorial: Generative Adversarial Networks". In: *CoRR* abs/1701.00160 (2017), pp. 44–46.

[19] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron C. Courville, and Yoshua Bengio. "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, Montréal, QC, Canada, December 8-13*. 2014, pp. 2672–2680.

[20] André Grüning and Sander M. Bohte. "Spiking Neural Networks: Principles and Challenges". In: *22th European Symposium on Artificial Neural Networks, ESANN 2014, Bruges, Belgium, April 23-25*. 2014.

[21] Ishaan Gulrajani, Faruk Ahmed, Martín Arjovsky, Vincent Dumoulin, and Aaron C. Courville. "Improved Training of Wasserstein GANs". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, December 4-9*. 2017, pp. 5767–5777.

[22] Tanmay Gupta, Dustin Schwenk, Ali Farhadi, Derek Hoiem, and Aniruddha Kembhavi. “Imagine This! Scripts to Compositions to Videos”. In: *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, Proceedings, Part VIII*. Vol. 11212. Lecture Notes in Computer Science. 2018, pp. 610–626.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep Residual Learning for Image Recognition”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30*. IEEE Computer Society, 2016, pp. 770–778.

[24] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”. In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, December 4-9*. 2017, pp. 6626–6637.

[25] Sepp Hochreiter. “The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions”. In: *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 6.2 (1998), pp. 107–116.

[26] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (1997), pp. 1735–1780.

[27] Jeremy Howard and Sebastian Ruder. “Universal Language Model Fine-tuning for Text Classification”. In: *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, Volume 1: Long Papers*. 2018, pp. 328–339.

[28] Jie Hu, Li Shen, and Gang Sun. “Squeeze-and-Excitation Networks”. In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22*. IEEE Computer Society, 2018, pp. 7132–7141.

[29] Sergey Ioffe and Christian Szegedy. “Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift”. In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, July 6-11*. Vol. 37. JMLR Workshop and Conference Proceedings. JMLR.org, 2015, pp. 448–456.

[30] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. “Image-to-Image Translation with Conditional Adversarial Networks”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26*. 2017, pp. 5967–5976.

[31] Saumya Jetley, Nicholas A. Lord, Namhoon Lee, and Philip H. S. Torr. “Learn to Pay Attention”. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. 2018.

[32] Justin Johnson, Agrim Gupta, and Li Fei-Fei. “Image Generation From Scene Graphs”. In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22*. 2018, pp. 1219–1228.

[33] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, Conference Track Proceedings*. 2015.

[34] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, Conference Track Proceedings*. 2014.

[35] Tejas D. Kulkarni, Pushmeet Kohli, Joshua B. Tenenbaum, and Vikash K. Mansinghka. "Picture: A probabilistic programming language for scene perception". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12*. 2015, pp. 4390–4399.

[36] Tejas D. Kulkarni, William F. Whitney, Pushmeet Kohli, and Joshua B. Tenenbaum. "Deep Convolutional Inverse Graphics Network". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montréal, QC, Canada, December 7-12*. 2015, pp. 2539–2547.

[37] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[38] Bowen Li, Xiaojuan Qi, Thomas Lukasiewicz, and Philip H. S. Torr. "Controllable Text-to-Image Generation". In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, BC, Canada, December 8-14*. 2019, pp. 2063–2073.

[39] Zhuorong Li, Minghui Wu, Jianwei Zheng, and Hongchuan Yu. "Perceptual Adversarial Networks With a Feature Pyramid for Image Translation". In: *IEEE Computer Graphics and Applications* 39.4 (2019), pp. 68–77.

[40] Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. "Are GANs Created Equal? A Large-Scale Study". In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, Montréal, QC, Canada, December 3-8*. 2018, pp. 698–707.

[41] Andrew L Maas, Awni Y Hannun, and Andrew Y Ng. "Rectifier nonlinearities improve neural network acoustic models". In: *Proc. icml*. Vol. 30. 1. 2013, p. 3.

[42] Wolfgang Maass. "Networks of spiking neurons: The third generation of neural network models". In: *Neural Networks* 10.9 (1997), pp. 1659–1671.

[43] Masoud MahdianPari, Bahram Salehi, Mohammad Rezaee, Fariba Mohammadimanesh, and Yun Zhang. "Very Deep Convolutional Neural Networks for Complex Land Cover Mapping Using Multispectral Remote Sensing Imagery". In: *Remote. Sens.* 10.7 (2018), p. 1119.

[44] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. "Spectral Normalization for Generative Adversarial Networks". In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, Conference Track Proceedings*. OpenReview.net, 2018.

[45] Augustus Odena, Christopher Olah, and Jonathon Shlens. “Conditional Image Synthesis with Auxiliary Classifier GANs”. In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, August 6-11*. Vol. 70. Proceedings of Machine Learning Research. PMLR, 2017, pp. 2642–2651.

[46] Aäron van den Oord, Nal Kalchbrenner, Lasse Espeholt, Koray Kavukcuoglu, Oriol Vinyals, and Alex Graves. “Conditional Image Generation with PixelCNN Decoders”. In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, Barcelona, Spain, December 5-10*. 2016, pp. 4790–4798.

[47] Aäron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. “Pixel Recurrent Neural Networks”. In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24*. Vol. 48. JMLR Workshop and Conference Proceedings. 2016, pp. 1747–1756.

[48] Niki Parmar, Prajit Ramachandran, Ashish Vaswani, Irwan Bello, Anselm Levskaya, and Jon Shlens. “Stand-Alone Self-Attention in Vision Models”. In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems, NeurIPS 2019, Vancouver, BC, Canada December 8-14*. 2019, pp. 68–80.

[49] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. “Deep Contextualized Word Representations”. In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, LA, USA, June 1-6, Volume 1 (Long Papers)*. 2018, pp. 2227–2237.

[50] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016.

[51] Guo-Jun Qi. “Loss-Sensitive Generative Adversarial Networks on Lipschitz Densities”. In: *CoRR* abs/1701.06264 (2017).

[52] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. “Learn, Imagine and Create: Text-to-Image Generation from Prior Knowledge”. In: *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, Vancouver, BC, Canada, December 8-14*. 2019, pp. 885–895.

[53] Tingting Qiao, Jing Zhang, Duanqing Xu, and Dacheng Tao. “MirrorGAN: Learning Text-To-Image Generation by Redescription”. In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20*. Computer Vision Foundation / IEEE, 2019, pp. 1505–1514.

[54] Alec Radford, Luke Metz, and Soumith Chintala. “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”. In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, Conference Track Proceedings*. 2016.

[55] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. "Improving language understanding by generative pre-training". In: *URL https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf* (2018).

[56] Scott E. Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, and Honglak Lee. "Learning What and Where to Draw". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, Barcelona, Spain, December 5-10*. 2016, pp. 217–225.

[57] Scott E. Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. "Generative Adversarial Text to Image Synthesis". In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24*. Vol. 48. JMLR Workshop and Conference Proceedings. 2016, pp. 1060–1069.

[58] Scott E. Reed, Aäron van den Oord, Nal Kalchbrenner, Sergio Gomez Colmenarejo, Ziyu Wang, Yutian Chen, Dan Belov, and Nando de Freitas. "Parallel Multiscale Autoregressive Density Estimation". In: *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, August 6-11*. Vol. 70. Proceedings of Machine Learning Research. 2017, pp. 2912–2921.

[59] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "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*. Vol. 9351. Lecture Notes in Computer Science. 2015, pp. 234–241.

[60] Mihaela Rosca, Balaji Lakshminarayanan, David Warde-Farley, and Shakir Mohamed. "Variational Approaches for Auto-Encoding Generative Adversarial Networks". In: *CoRR* abs/1706.04987 (2017).

[61] Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. "Improved Techniques for Training GANs". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, Barcelona, Spain, December 5-10*. 2016, pp. 2226–2234.

[62] Shibani Santurkar, Dimitris Tsipras, Andrew Ilyas, and Aleksander Madry. "How Does Batch Normalization Help Optimization?" In: *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, Montréal, QC, Canada, December 3-8*. 2018, pp. 2488–2498.

[63] Jo Schlemper, Ozan Oktay, Michiel Schaap, Mattias P. Heinrich, Bernhard Kainz, Ben Glocker, and Daniel Rueckert. "Attention gated networks: Learning to leverage salient regions in medical images". In: *Medical Image Analysis* 53 (2019), pp. 197–207.

[64] Jake Snell, Karl Ridgeway, Renjie Liao, Brett D. Roads, Michael C. Mozer, and Richard S. Zemel. "Learning to generate images with perceptual similarity metrics". In: *2017 IEEE International Conference on Image Processing, ICIP 2017, Beijing, China, September 17-20*. 2017, pp. 4277–4281.

[65] Karl Stratos, Do-kyum Kim, Michael Collins, and Daniel J. Hsu. "A Spectral Algorithm for Learning Class-Based n-gram Models of Natural Language". In: *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence, UAI 2014, Quebec City, QC, Canada, July 23-27*. 2014, pp. 762–771.

[66] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. "Rethinking the Inception Architecture for Computer Vision". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30*. IEEE Computer Society, 2016, pp. 2818–2826.

[67] Lucas Theis and Matthias Bethge. "Generative Image Modeling Using Spatial LSTMs". In: *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, Montréal, QC, Canada, December 7-12*. 2015, pp. 1927–1935.

[68] Lucas Theis, Aäron van den Oord, and Matthias Bethge. "A note on the evaluation of generative models". In: *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, Conference Track Proceedings*. 2016.

[69] Masatoshi Uehara, Issei Sato, Masahiro Suzuki, Kotaro Nakayama, and Yutaka Matsuo. "Generative adversarial nets from a density ratio estimation perspective". In: *arXiv preprint arXiv:1610.02920* (2016).

[70] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. "Attention is All you Need". In: *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, Long Beach, CA, USA, December 4-9*. 2017, pp. 6000–6010.

[71] Andreas Veit, Michael J. Wilber, and Serge J. Belongie. "Residual Networks Behave Like Ensembles of Relatively Shallow Networks". In: *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, Barcelona, Spain, December 5-10, 2016*. 2016, pp. 550–558.

[72] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. "Extracting and composing robust features with denoising autoencoders". In: *Machine Learning, Proceedings of the Twenty-Fifth International Conference (ICML 2008), Helsinki, Finland, June 5-9*. Vol. 307. ACM International Conference Proceeding Series. 2008, pp. 1096–1103.

[73] Jilles Vreeken. *Spiking Neural Networks, an Introduction*. Tech. rep. 2003.

[74] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. "The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001". In: (2011).

[75] Alexander H. Waibel, Toshiyuki Hanazawa, Geoffrey E. Hinton, Kiyohiro Shikano, and Kevin J. Lang. "Phoneme recognition using time-delay neural networks". In: *IEEE Trans. Acoust. Speech Signal Process.* 37.3 (1989), pp. 328–339.

[76] Z. Wang, E. P. Simoncelli, and A. C. Bovik. "Multiscale structural similarity for image quality assessment". In: *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*. Vol. 2. Nov. 2003, 1398–1402 Vol.2.

[77] Jiajun Wu, Joshua B. Tenenbaum, and Pushmeet Kohli. "Neural Scene De-rendering". In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[78] Jianwen Xie, Yang Lu, Song-Chun Zhu, and Ying Nian Wu. "A Theory of Generative ConvNet". In: *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24*. Vol. 48. JMLR Workshop and Conference Proceedings. 2016, pp. 2635–2644.

[79] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention". In: *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, July 6-11*. Vol. 37. JMLR Workshop and Conference Proceedings. 2015, pp. 2048–2057.

[80] Qiantong Xu, Gao Huang, Yang Yuan, Chuan Guo, Yu Sun, Felix Wu, and Kilian Q. Weinberger. "An empirical study on evaluation metrics of generative adversarial networks". In: *CoRR* abs/1806.07755 (2018).

[81] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. "AttnGAN: Fine-Grained Text to Image Generation With Attentional Generative Adversarial Networks". In: *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22*. IEEE Computer Society, 2018, pp. 1316–1324.

[82] Guojun Yin, Bin Liu, Lu Sheng, Nenghai Yu, Xiaogang Wang, and Jing Shao. "Semantics Disentangling for Text-To-Image Generation". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20*. Computer Vision Foundation / IEEE, 2019, pp. 2327–2336.

[83] Han Zhang, Ian J. Goodfellow, Dimitris N. Metaxas, and Augustus Odena. "Self-Attention Generative Adversarial Networks". In: *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, Long Beach, California, USA, June 9-15*. Vol. 97. Proceedings of Machine Learning Research. PMLR, 2019, pp. 7354–7363.

[84] Han Zhang, Tao Xu, and Hongsheng Li. "StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks". In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29*. 2017, pp. 5908–5916.

[85] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. "StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks". In: *CoRR* abs/1710.10916 (2017).

[86] Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang. "DM-GAN: Dynamic Memory Generative Adversarial Networks for Text-To-Image Synthesis". In: *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20*. Computer Vision Foundation / IEEE, 2019, pp. 5802–5810.

[87] Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. “Aligning Books and Movies: Towards Story-Like Visual Explanations by Watching Movies and Reading Books”. In: *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13*. 2015, pp. 19–27.