Hudson & Zitnick

GANsformer reproducibility challenge

Project for Advance Topic in Machine Learning course (21/22)

https://github.com/GiorgiaAuroraAdorni/gansformer-reproducibility-challenge

Giorgia Adorni
giorgia.adorni@usi.ch

Stefano Carlo Lambertenghi
stefano.carlo.lambertenghi@usi.ch

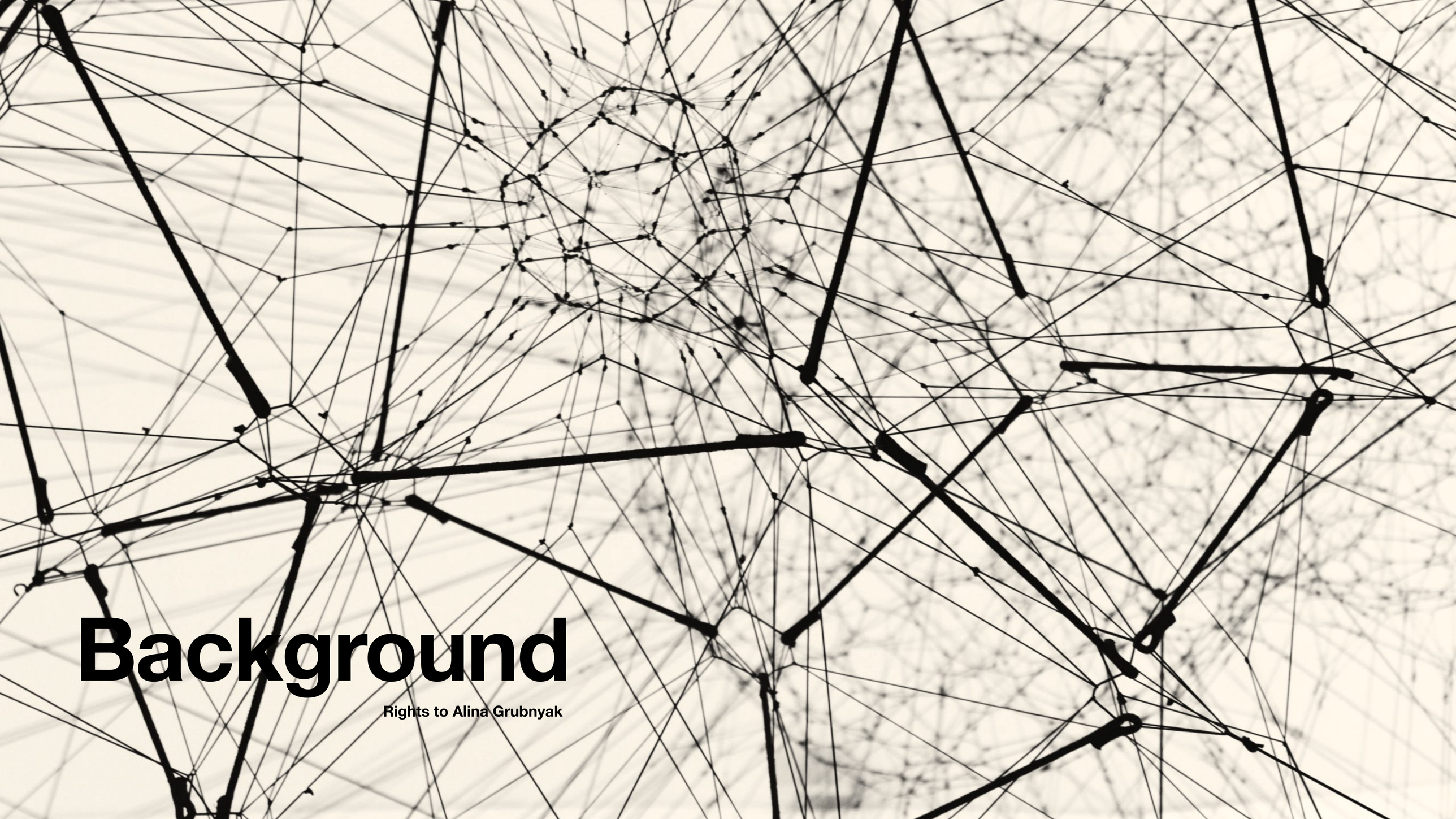
Felix Boelter
felix.boelter@usi.ch

Introduction

- Generative modelling is a “hot” topic nowadays used to generate unreal images
- We investigate the reliability and reproducibility of “Generative Adversarial Transformers” by Hudson and Zitnick [2]
  - State-of-the art results and faster learning
- Recreate the paper methods and reproduce their results
- Models are implemented using authors’ code and information
- We reproduced the GANformer with two types of attention and a StyleGAN2 as a baseline.
- Performed tests to validate the claimed results.
- Discuss inconsistencies and solutions. And finally perform a Demo.

Agenda

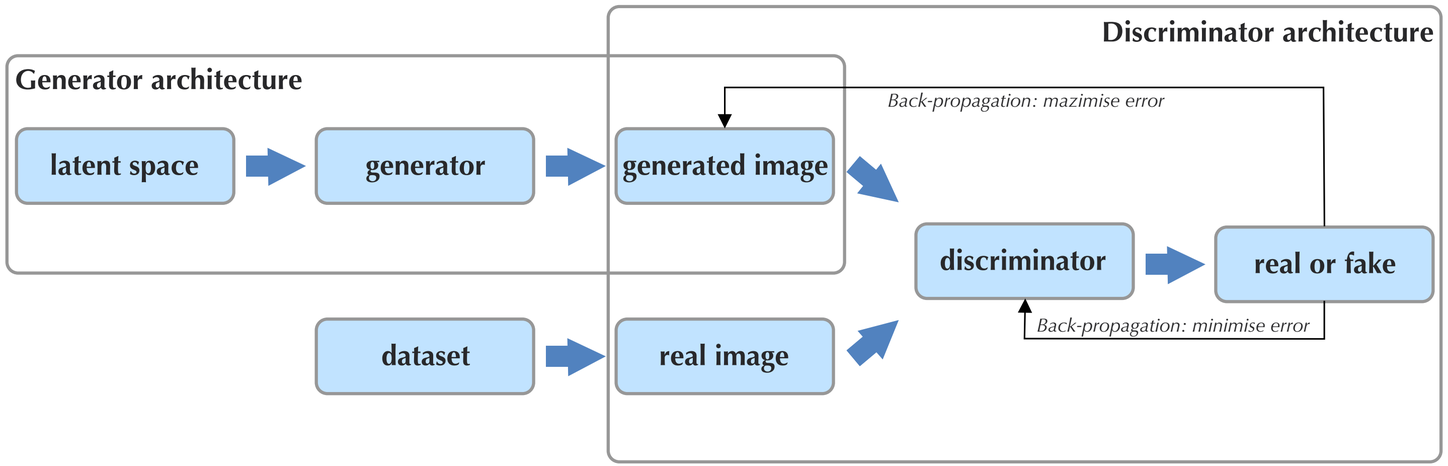
- Introduction
- Background
  - GANs
  - StyleGAN2
  - Attention
- Generative Adversarial Transformers
  - Simplex and Duplex attention
- Implementation
  - Datasets
  - Hyper-parameters and design choices
  - Experimental setup
  - Computational requirements
- Results
- Implementation issues
- Results discussion
- Conclusions



Background

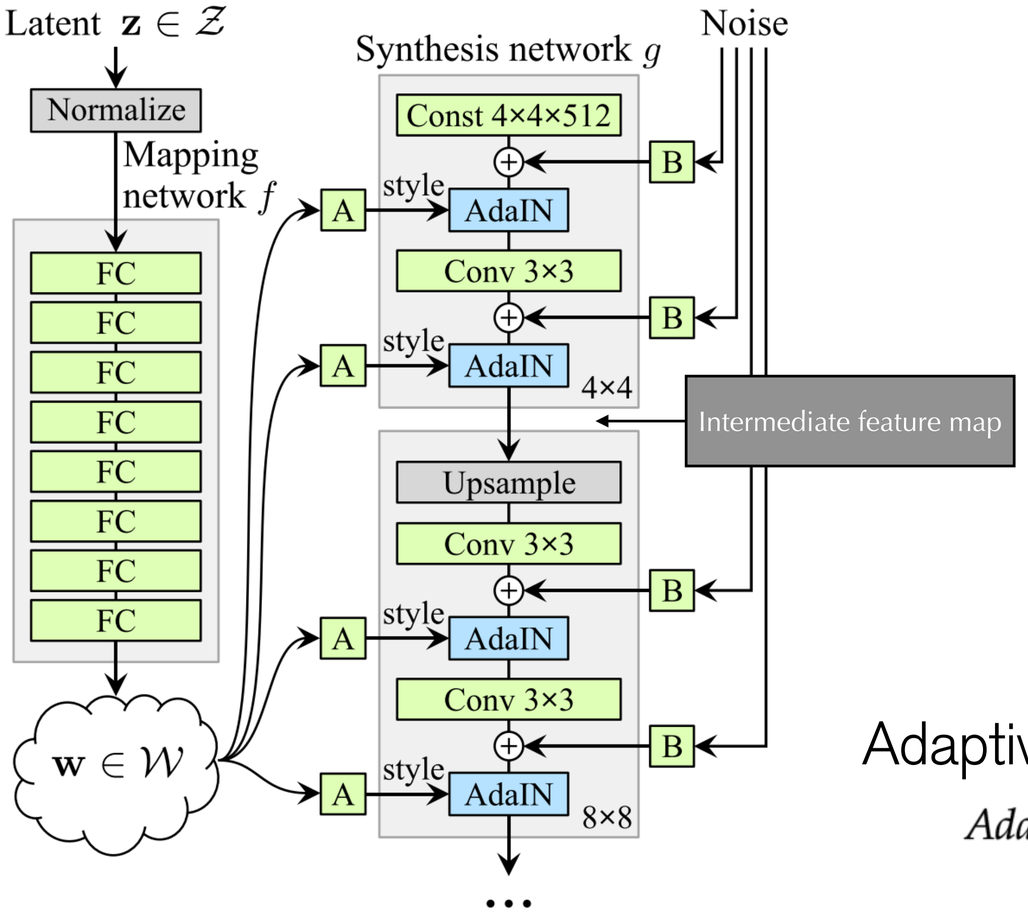
Generative Adversarial Networks (GANs) [1]

- GANs: deep-learning-based generative models
  - the generator G(z): generates new plausible examples in the problem domain → images
  - the discriminator D(x): classifies the examples as real (coming from the training dataset) or fake (generated by G)

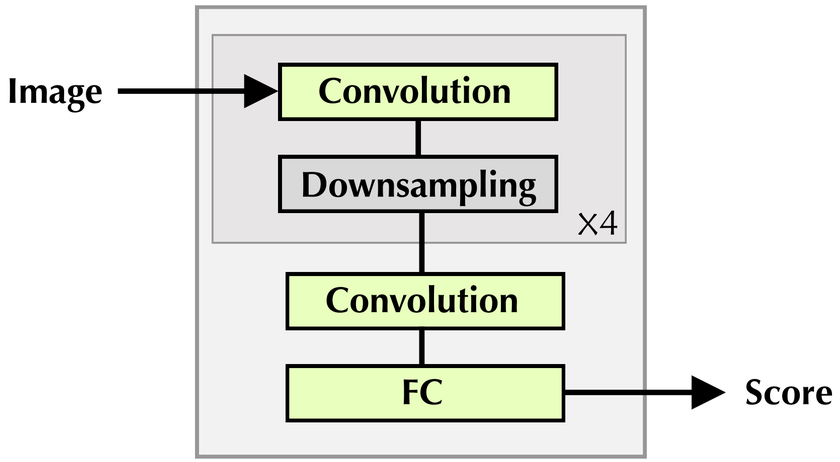


StyleGAN [4]

Generator



Discriminator

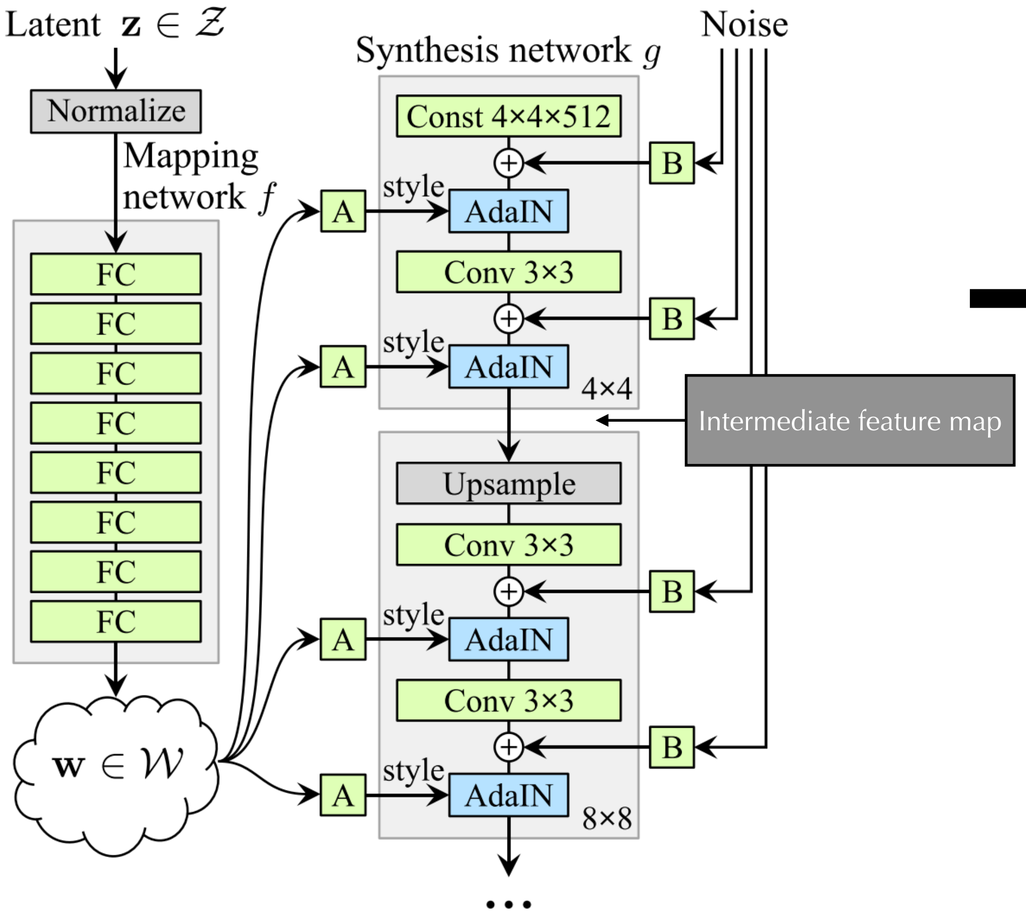


Adaptive Instance Normalisation

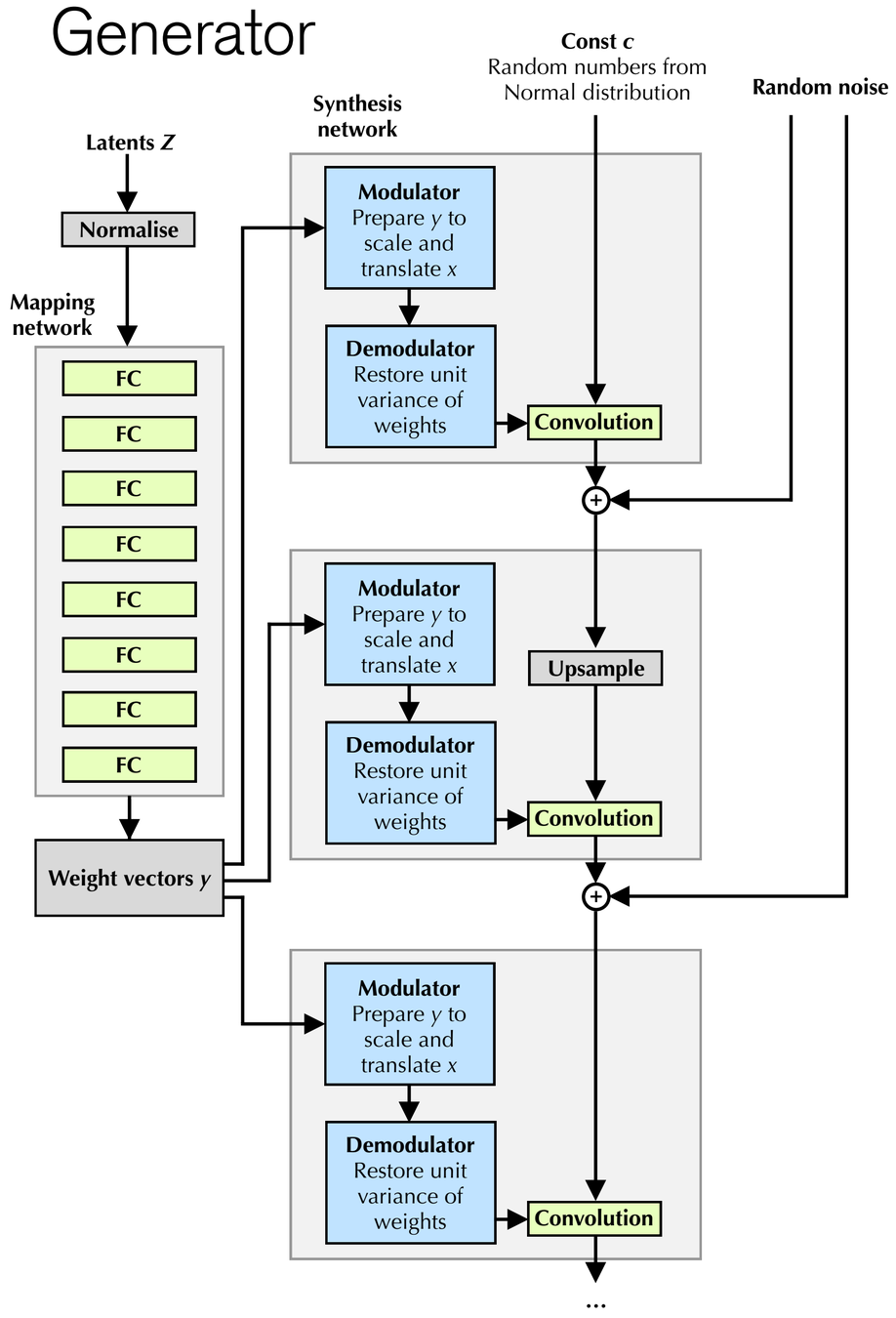
$$AdaIn(x, y) = \sigma(y)\left(\frac{x-\mu(x)}{\sigma(x)}\right) + \mu(y)$$

StyleGAN2 [3]

Generator



Generator



~~Transformers~~ Attention is all you need! [5, 6]

- Multi-layer bidirectional transformer encoder (BERT)
- The attention mechanism lets the network understand which are the image features by deciding on which parts of the image to concentrate on
- Pros:

  \+ Strength for long-range interactions → understand the context of the image

  \+ More parallelisation and less training times

- Cons:

  – High demand of computational resources


Generative Adversarial
Transformers

Generative Adversarial Transformers [2]

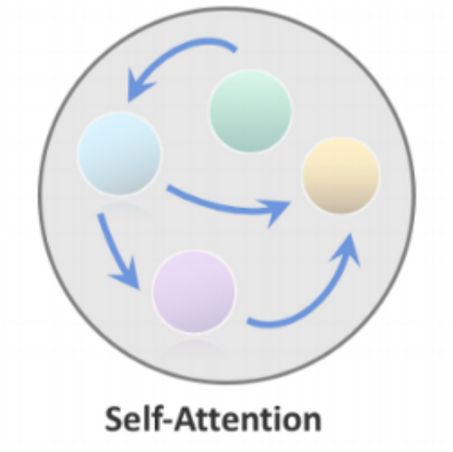
- Combination of StyleGAN2 and BERT attention mechanism to generate complex and more realistic examples

- Why StyleGAN2?
  - Powerful generator for the overall style of the image with control of global features.

- Why BERT attention mechanism?
  - Powerful with respect to small details of localised regions.

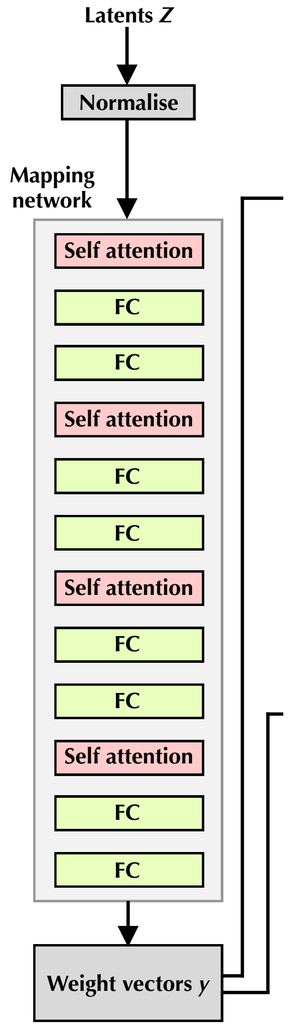
Generative Adversarial Transformers [2]

Standard transformer



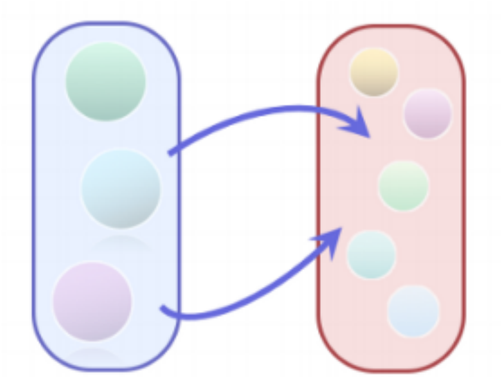
- GANformer uses self-attention only in the in the generator mapping network.
- Alternates multi-head self-attention and feed-forward layers.

Generator

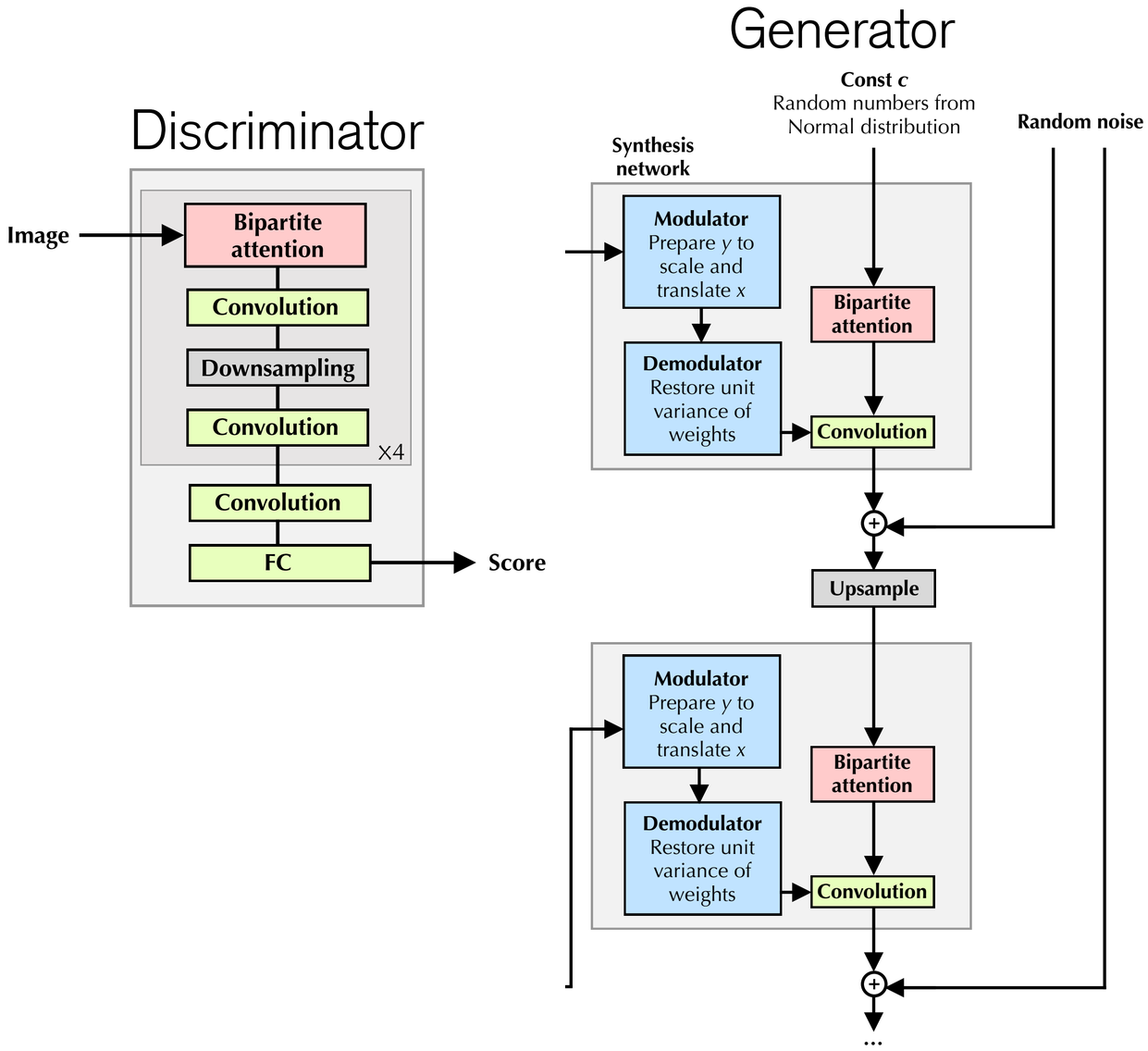


Generative Adversarial Transformers [2]

Bipartite transformer

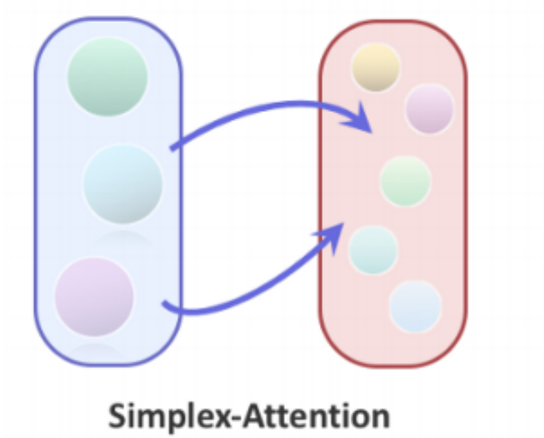


- Bipartite attention allows the network to decide which are the important image features using the latent variables.



Generative Adversarial Transformers [2]

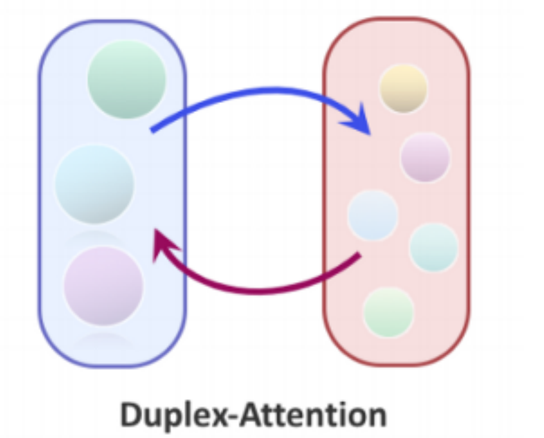
Simplex attention



- permits communication in a single direction:

  latent vectors → image features

Duplex attention

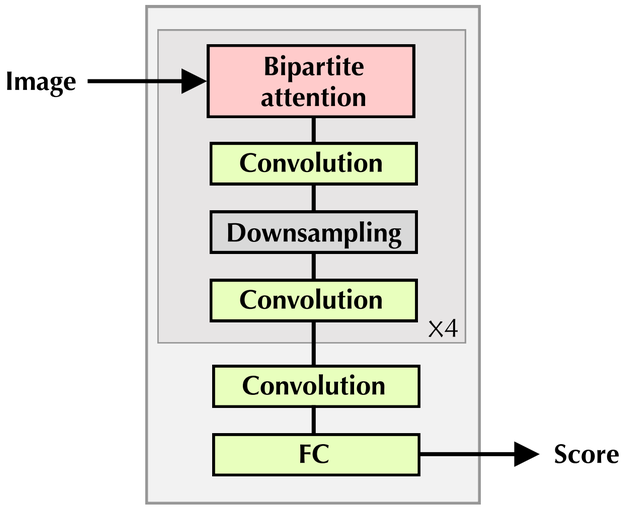


- permits communication in both directions:

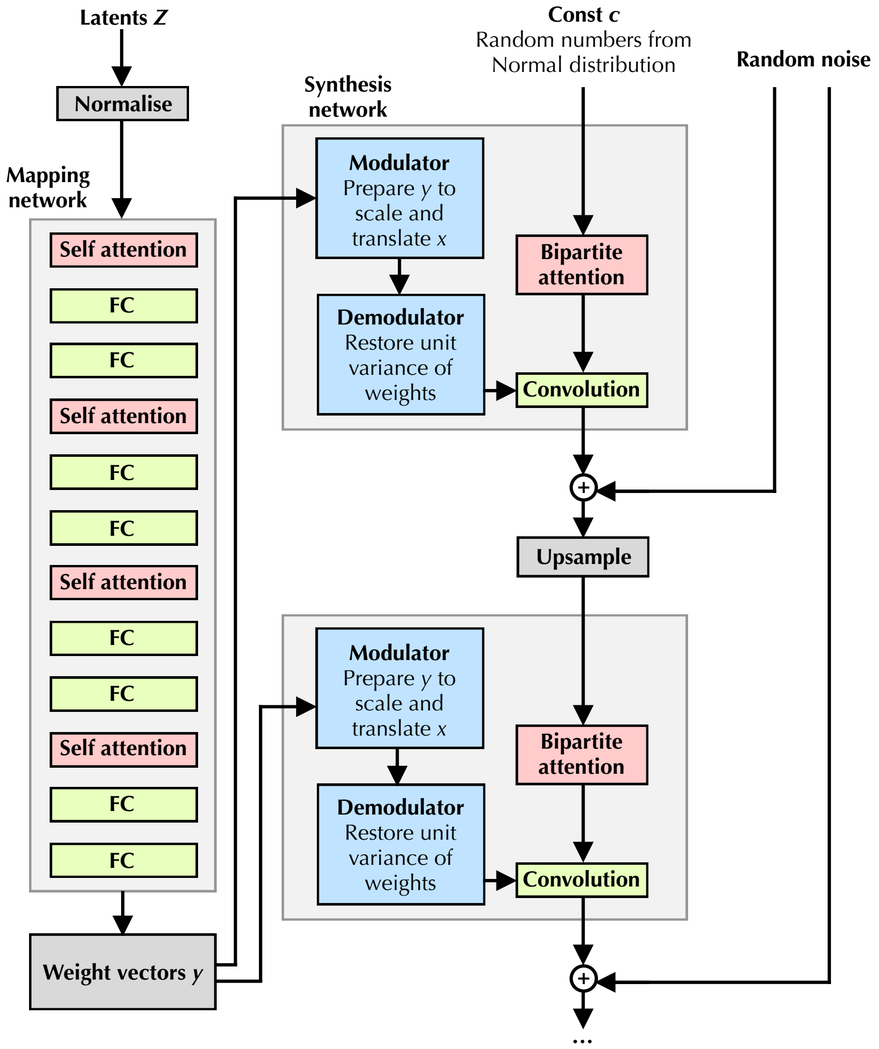
  first compute the attention assignments between image features and latent vectors and then refine the assignments of the image features by considering the centroids of the latents (k-means algorithm)

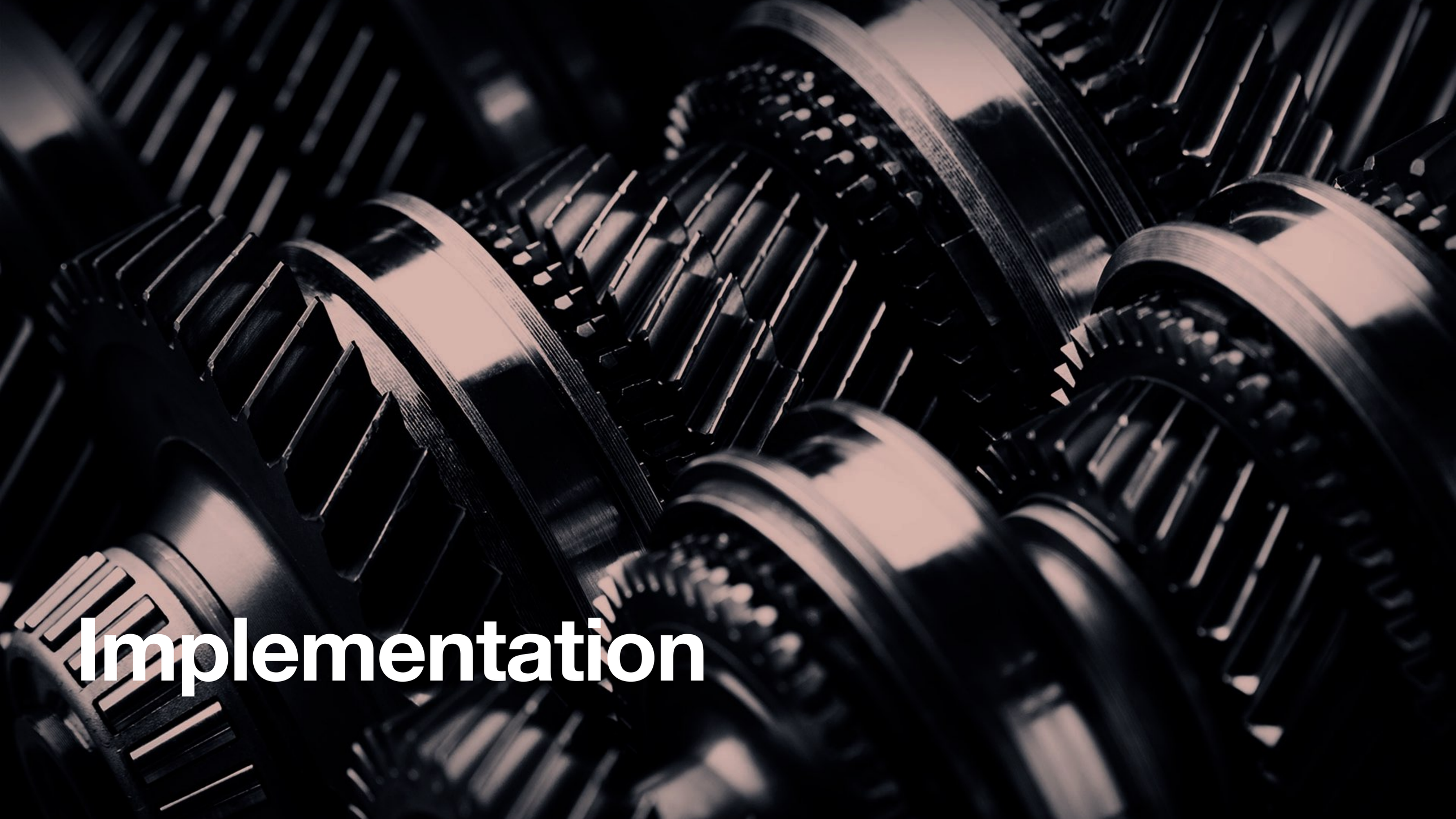
Generative Adversarial Transformers [2]

Discriminator



Generator





Implementation

Datasets: Original paper

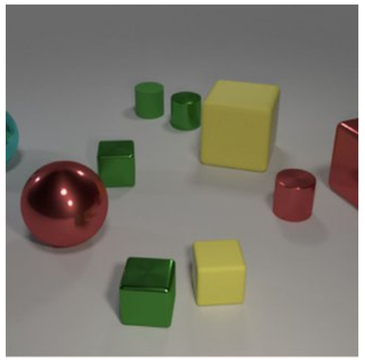
- FFHQ

Images: 70k

TFrecords size: 13 Gb

Resolution: 256x256

- CLEVR



Images: 100k

TFrecords size: 15.5 Gb

Resolution: 256x256

- LSUN-Bedrooms

Images: 3M

TFrecords size: 480 Gb

Resolution: 256x256

- Cityscapes

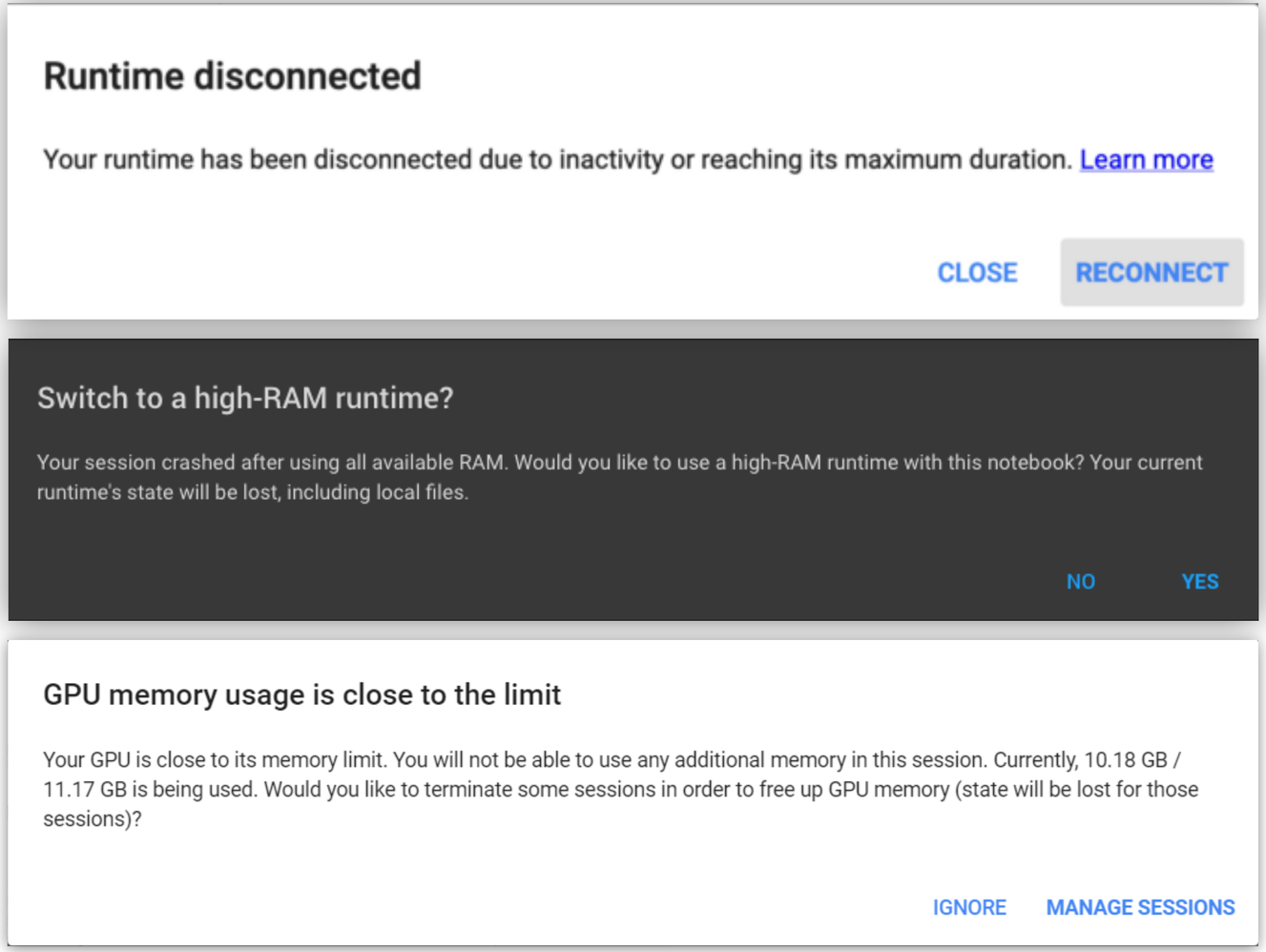
Images: 24k

TFrecords size: 8 Gb

Resolution: 256x256

Datasets: First attempt

Datasets: First attempt



Datasets: First attempt

Runtime disconnected

Your runtime has been disconnected due to inactivity or reaching its maximum duration. Learn more

CLOSE RECONNECT

Switch to a high-RAM runtime?

Your session crashed after using all available RAM. Would you like to use a high-RAM runtime with this notebook? Your current runtime's state will be lost, including local files.

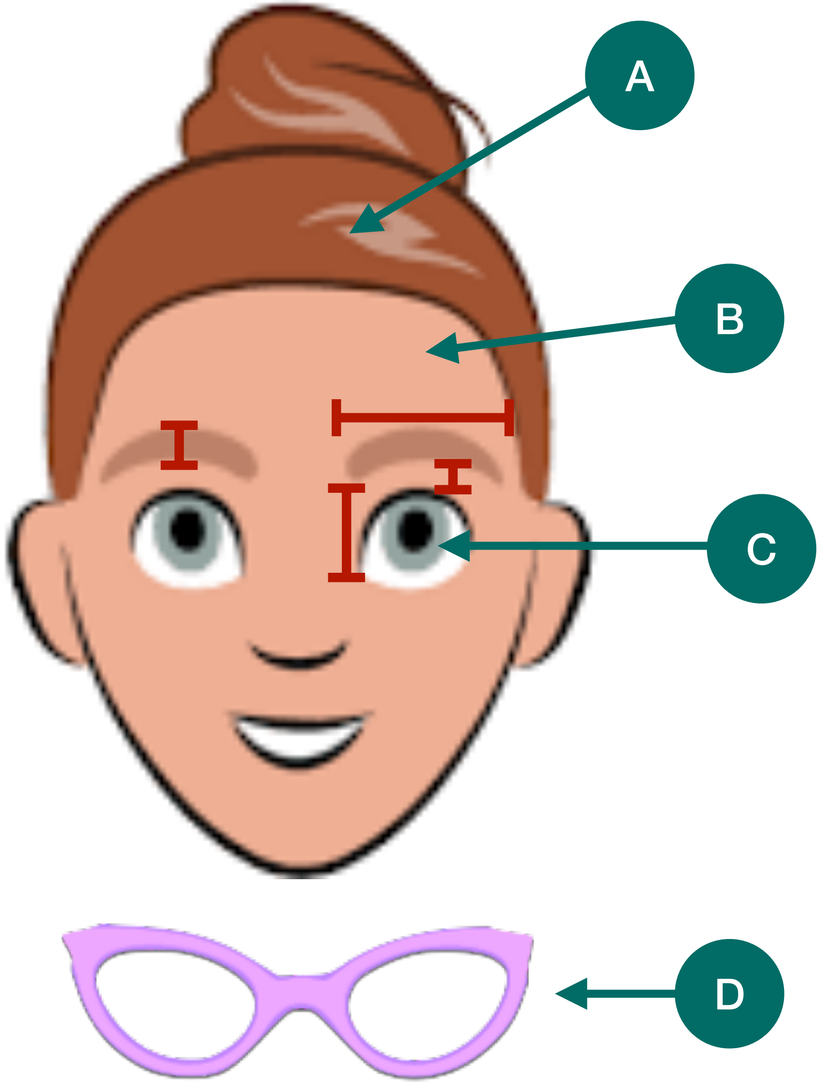
NO YES

GPU memory usage is close to the limit

Your GPU is close to its memory limit. You will not be able to use any additional memory in this session. Currently, 10.18 GB / 11.17 GB is being used. Would you like to terminate some sessions in order to free up GPU memory (state will be lost for those sessions)?

IGNORE MANAGE SESSIONS

Datasets: Cartoonset 10k



Dataset size (Tfrecords): 1.8 Gb

Number of images: 10k

Image size: 64 x 64 pixels

Type	Attribute	Number of variants
Artwork	Eyes	12
	Glasses	12
	Face shape	7
	Hair	111
	Facial hair	360
Colors	(A) Hair	5
	(B) Skin	11
	(C) Iris	7
	(D) Glasses	10
Proportions	4 attributes	108

Hyper-parameters

Adam Optimizer

- Epsilon

  Used to avoid division by zero

- Beta1 & Beta2

  Exponential decay rates for the moment estimates (mean and uncentered variance of gradients)

Mapping Network

- Component number

  Features of the generated images

- Latent size

  Size of the latent vector (convolution weights)

Hyper-parameters discrepancies

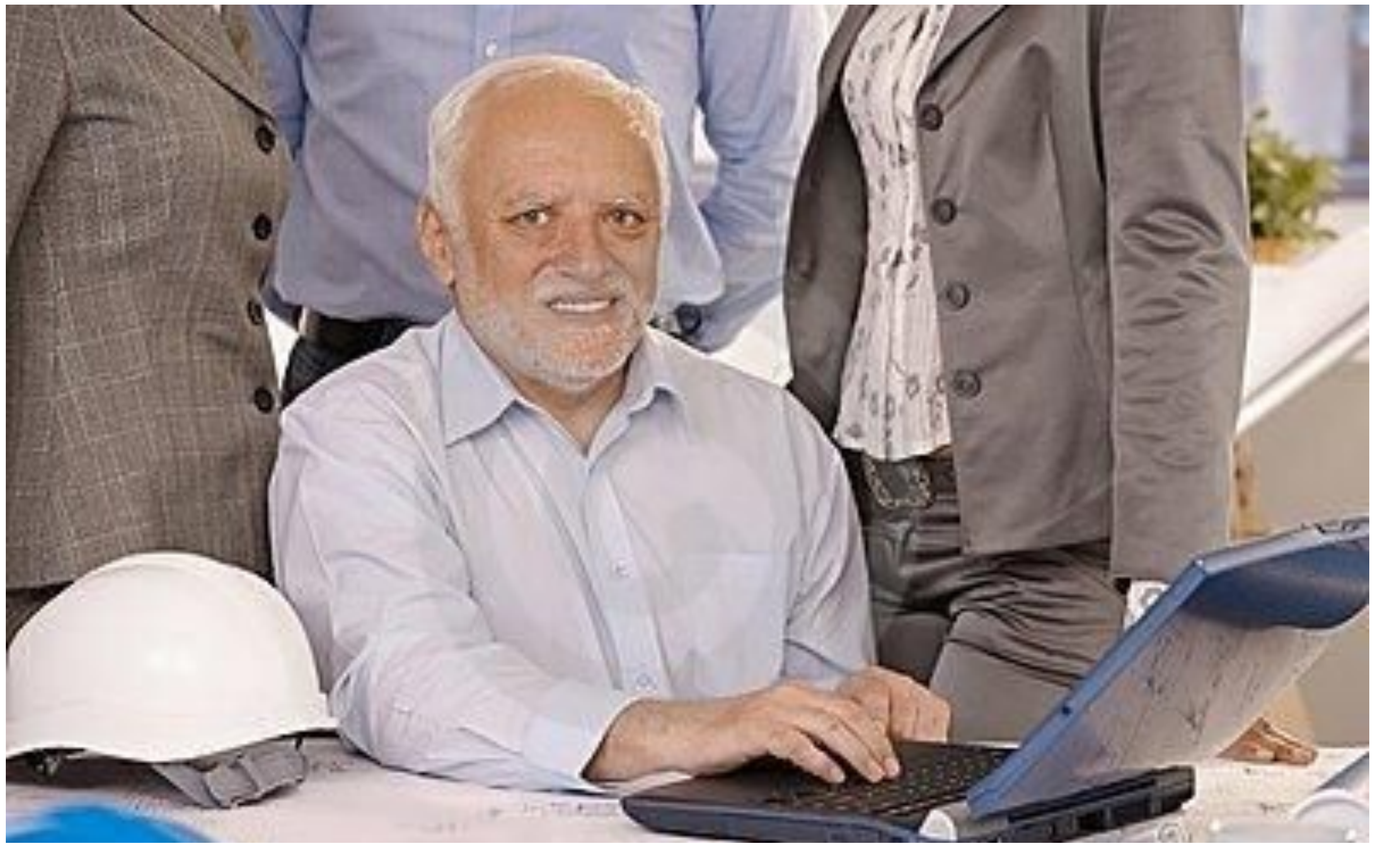
Paper's Hyperparameters

Model	StyleGAN2	GANformer$_s$	GANformer$_d$
Epsilon	1E-08	0.001	0.001
Latent size	512	32	32
Beta1	0.0	0.9	0.9
Beta2	0.99	0.999	0.999
Component Number	1	16	16

Paper's code Hyperparameters

Model	StyleGAN2	GANformer$_s$	GANformer$_d$
Epsilon	1E-08	1E-08	1E-08
Latent size	512	32	32
Beta1	0.0	0.0	0.0
Beta2	0.99	0.99	0.99
Component Number	1	16	16

Hyper-parameters



?

Hyper-parameters

Our Choice

Our Hyperparameters

Model	StyleGAN2	GANformer$_s$	GANformer$_d$
Epsilon	1E-08	1E-08	1E-08
Latent size	512	32	32
Beta1	0.0	0.0	0.0
Beta2	0.99	0.99	0.99
Component Number	1	16	16

Experimental setup & Computational requirements

Paper's Implementation

- Used four NVIDIA V100 GPUs per model.
- Final results shown after 10,000K img samples.
- Implemented in Python using TensorFlow.
- Images set to 256x256 resolution.
- Training for one model took 3-4 days.

Our implementation

- Used one Tesla P100-PCIE-16GB GPU per model.
- Final results shown after 300K img samples.
- Implemented in Python using TensorFlow.
- Images set to 64x64 resolution
- Training for one model took 8-10 hours.

Results

Metrics

- FID (Frechet Inception Distance)

  Distance between feature distributions of real and fake images

- IS (Inception Score)

  Formalisation of the concept of “Realism” and “Diversity” only using generated images

- Precision

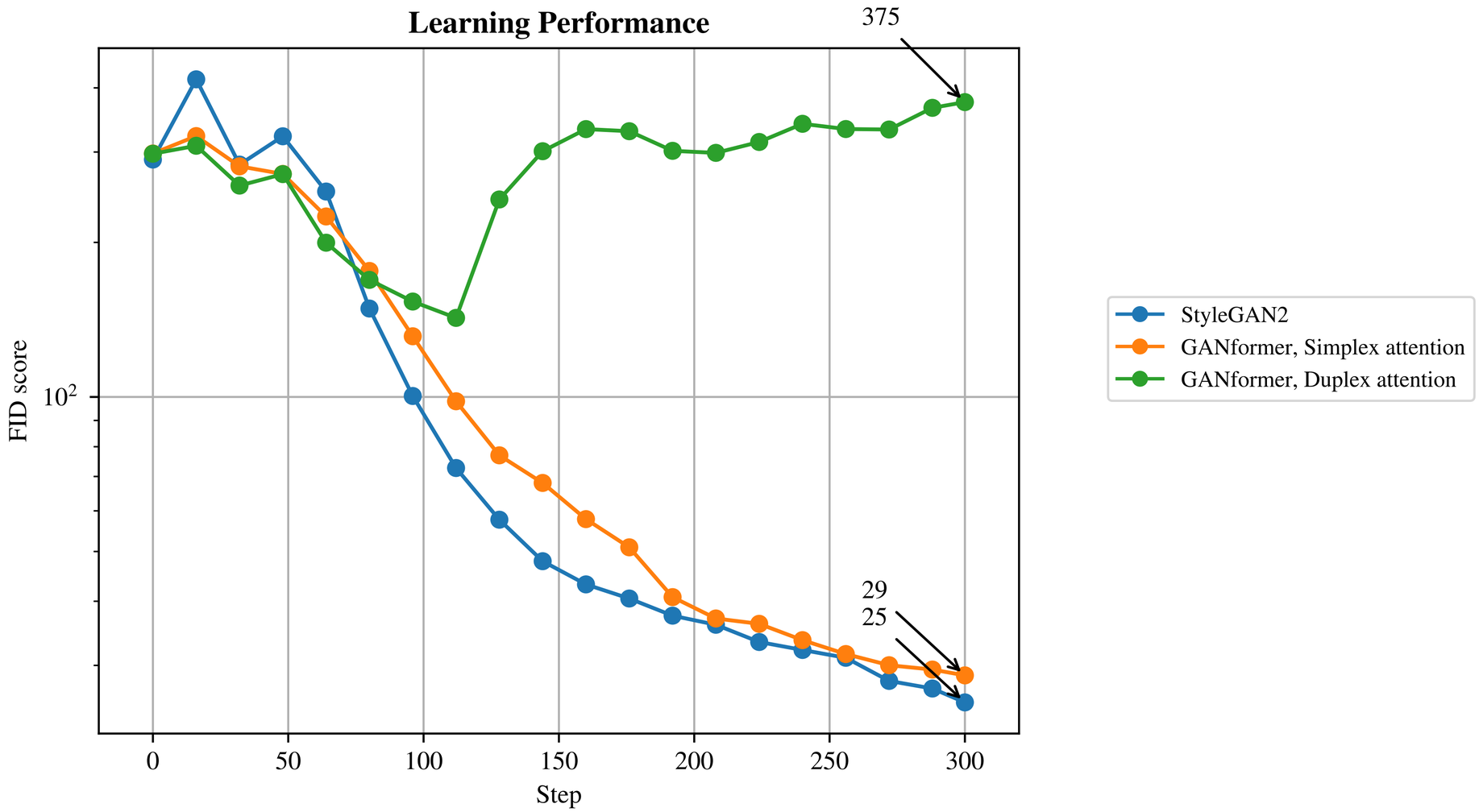
  Average similarity to the original images

- Recall

  Ability to generate any sample in the dataset

- Training time

Analysis of FID scores



Overall summary

Paper's Results

Model	StyleGAN2	GANformer$_s$	GANformer$_d$
FID ↓	11.29	10.29	7.22
IS ↑	2.74	2.82	2.78
Precison ↑	52.02	56.76	55.45
Recall ↑	23.98	18.21	33.94
FID Improvement Over baseline	0%	8.86%	36.11%
k-img/s	Not given	Not given	Not given

Our results

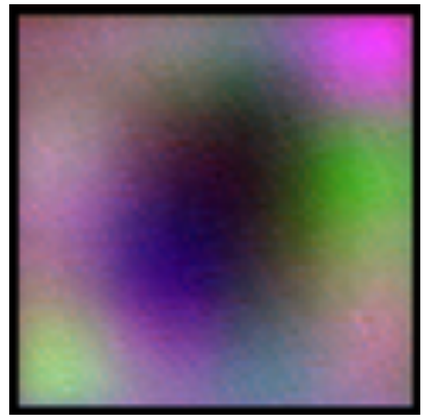
Model	StyleGAN2	GANformer$_s$	GANformer$_d$
FID ↓	24.77	28.11	374.18
IS ↑	2.50	2.58	1.40
Precison ↑	0.18	0.15	0
Recall ↑	2.11	0.76	0
FID Improvement Over baseline	0%	-13.47%	-1410.47%
k-img/s	91	~125	~125

Generated images

Dataset

StyleGAN2

GANformer
Simplex
Attention

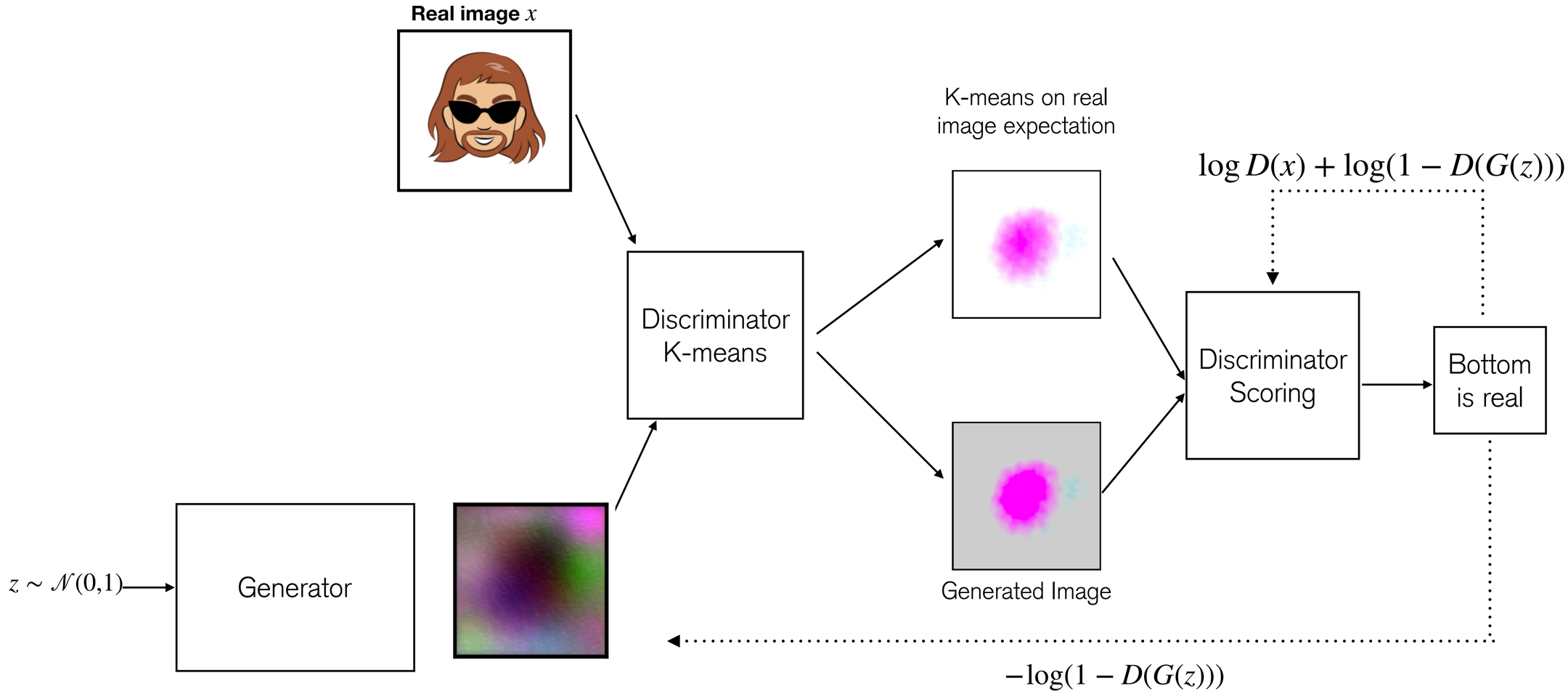


GANformer
Duplex
Attention

Results


Disappointing
Results

Why we think this is happening?



Why we think this is happening?

- K-means takes regions of the image and will compute centroid/mean of the combined regions.
- It uses the centroid/mean as a representative of those regions.
- This will turn the real image into something that doesn't look like the cartoon face.
- The Generator will try to create those images to try and fool the Discriminator.

Results

So how did they do it?

Pre-Trained model investigations

Expected Discriminator

D	Params	OutputShape	WeightShape
ltnt_emb/emb	512	(16, 32)	(16, 32)
64x64/FromRGB	2048	(?, 512, 64, 64)	(1, 1, 3, 512)
64x64/AttLayer_n2l	37056	(?, 32)	-
64x64/Conv0	2359808	(?, 512, 64, 64)	(3, 3, 512, 512)
64x64/Conv1_down	2359808	(?, 512, 32, 32)	(3, 3, 512, 512)
64x64/Skip	262144	(?, 512, 32, 32)	(1, 1, 512, 512)
32x32/AttLayer_n2l	37056	(?, 32)	-
32x32/Conv0	2359808	(?, 512, 32, 32)	(3, 3, 512, 512)
32x32/Conv1_down	2359808	(?, 512, 16, 16)	(3, 3, 512, 512)
32x32/Skip	262144	(?, 512, 16, 16)	(1, 1, 512, 512)
16x16/AttLayer_n2l	37056	(?, 32)	-
16x16/Conv0	2359808	(?, 512, 16, 16)	(3, 3, 512, 512)
16x16/Conv1_down	2359808	(?, 512, 8, 8)	(3, 3, 512, 512)
16x16/Skip	262144	(?, 512, 8, 8)	(1, 1, 512, 512)
8x8/AttLayer_n2l	37056	(?, 32)	-
8x8/Conv0	2359808	(?, 512, 8, 8)	(3, 3, 512, 512)
8x8/Conv1_down	2359808	(?, 512, 4, 4)	(3, 3, 512, 512)
8x8/Skip	262144	(?, 512, 4, 4)	(1, 1, 512, 512)
4x4/Conv	2364416	(?, 512, 4, 4)	(3, 3, 513, 512)
4x4/Dense0	4194816	(?, 512)	(8192, 512)
ComponentScores/Dense	1088	(?, 32)	(33, 32)
ComponentScores/Output	33	(?, 1)	(32, 1)
Output/weight	528	(528, 1)	(528, 1)
Output/bias	1	(1,)	(1,)
---	---	---	---
Total	26638706		

Found Discriminator

D	Params	OutputShape	WeightShape
64x64/FromRGB	2048	(?, 512, 64, 64)	(1, 1, 3, 512)
64x64/Conv0	2359808	(?, 512, 64, 64)	(3, 3, 512, 512
64x64/Conv1_down	2359808	(?, 512, 32, 32)	(3, 3, 512, 512
64x64/Skip	262144	(?, 512, 32, 32)	(1, 1, 512, 512
32x32/Conv0	2359808	(?, 512, 32, 32)	(3, 3, 512, 512
32x32/Conv1_down	2359808	(?, 512, 16, 16)	(3, 3, 512, 512
32x32/Skip	262144	(?, 512, 16, 16)	(1, 1, 512, 512
16x16/Conv0	2359808	(?, 512, 16, 16)	(3, 3, 512, 512
16x16/Conv1_down	2359808	(?, 512, 8, 8)	(3, 3, 512, 512
16x16/Skip	262144	(?, 512, 8, 8)	(1, 1, 512, 512
8x8/Conv0	2359808	(?, 512, 8, 8)	(3, 3, 512, 512
8x8/Conv1_down	2359808	(?, 512, 4, 4)	(3, 3, 512, 512
8x8/Skip	262144	(?, 512, 4, 4)	(1, 1, 512, 512
4x4/Conv	2364416	(?, 512, 4, 4)	(3, 3, 513, 512
4x4/Dense0	4194816	(?, 512)	(8192, 512)
Output/weight	512	(512, 1)	(512, 1)
Output/bias	1	(1,)	(1,)
---	---	---	---
Total	26488833		

Code investigations

```
# GANformer
cset(cG.args, "transformer", args.transformer)
cset(cD.args, "transformer", args.d_transformer)
```

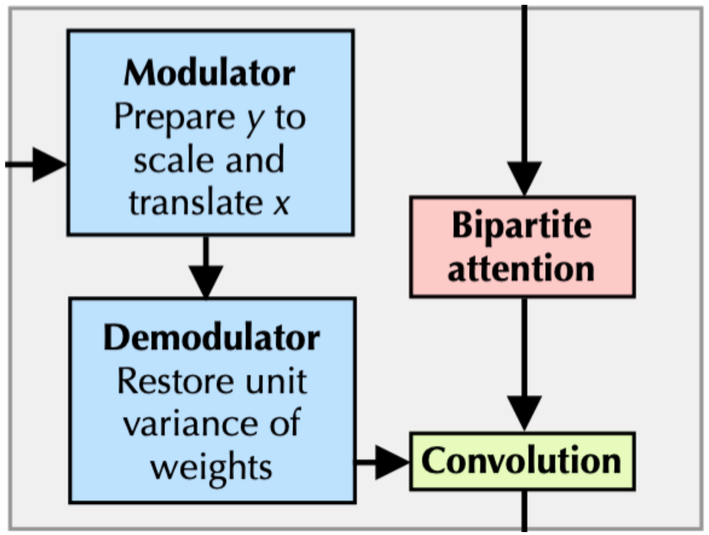
```
parser.add_argument("--d-transformer",      help = "Add transformer layers to the discriminator (bottom-up image-to-latents) (default: False)"
```

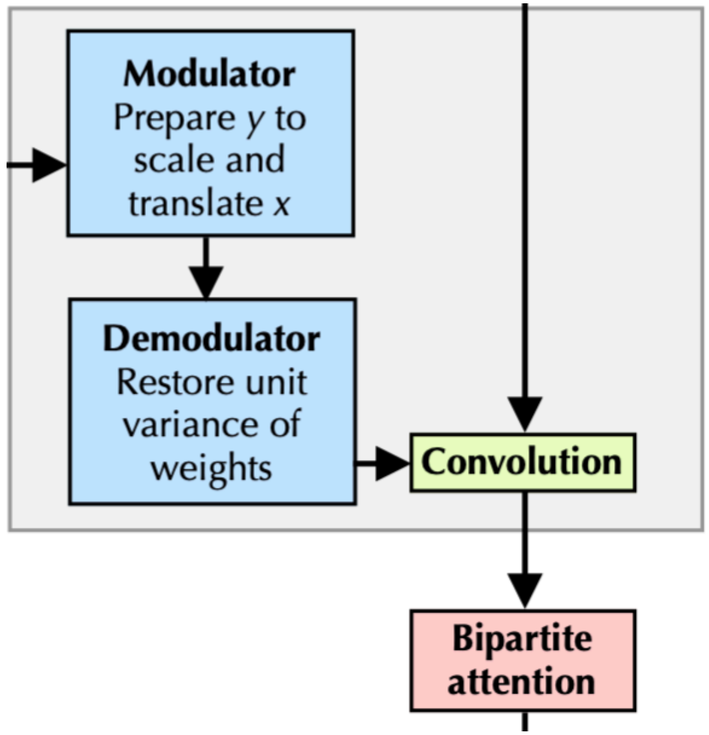
The authors never change the d-transformer argument!

Code investigations

Bipartite attention position

Paper



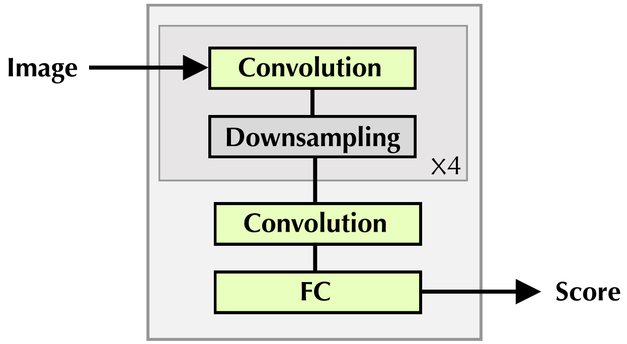


Code

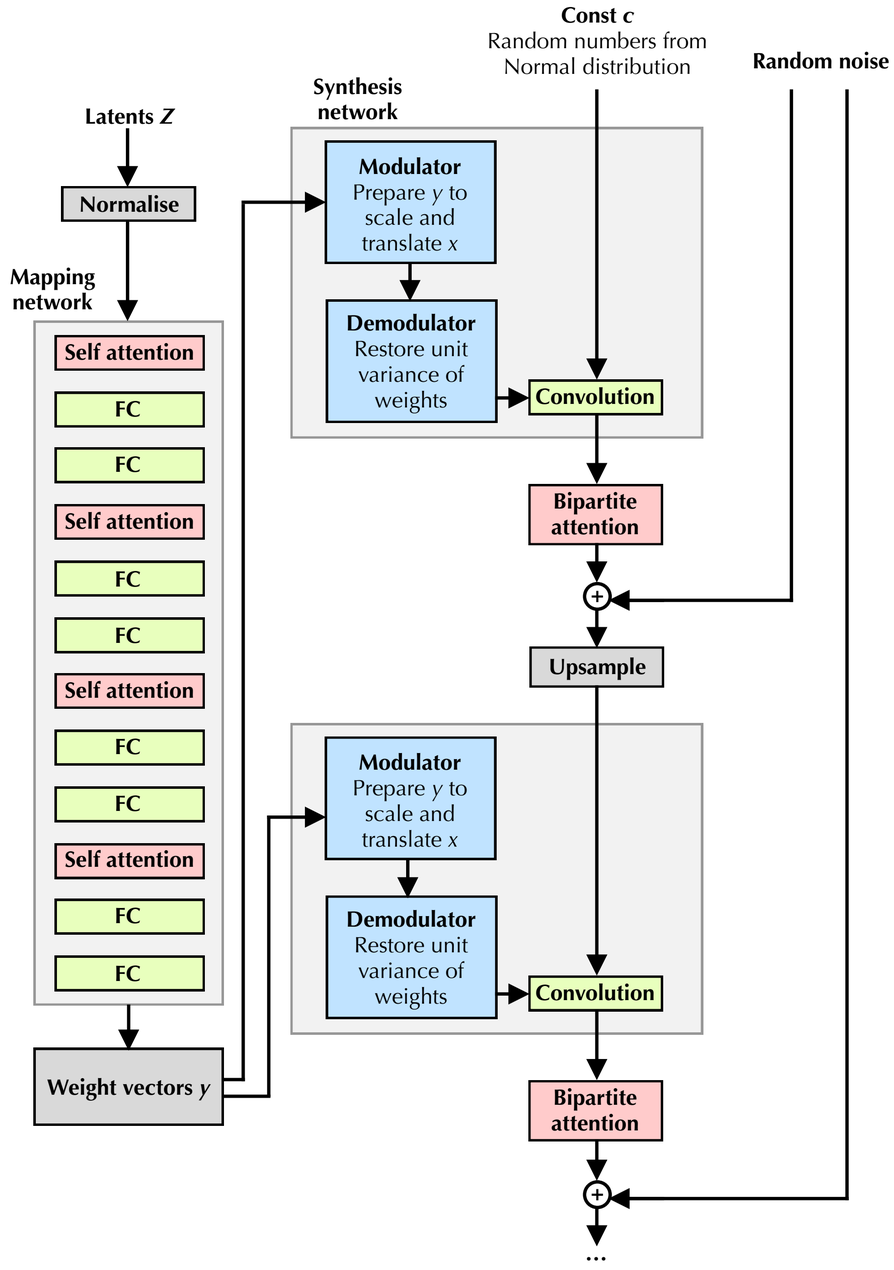
Our solution:
StyleGAN2/GANformer hybrids

New implementation

Discriminator



Generator



New overall summary

Previous results

Model	StyleGAN2	GANformer$_s$	GANformer$_d$
FID ↓	24.77	28.11	374.18
IS ↑	2.50	2.58	1.40
Precison ↑	0.18	0.15	0
Recall ↑	2.11	0.76	0
FID Improvement Over baseline	0%	-13.47%	-1410.47%
k-img/s	91	~125	~125

New results

Model	StyleGAN2	GANformer$_s$ Stylegan2 Discriminator	GANformer$_d$ Stylegan2 Discriminator
FID ↓	24.77	19.09	24.81
IS ↑	2.50	2.62	2.62
Precison ↑	0.18	0.35	0.35
Recall ↑	2.11	4.76	2.11
FID Improvement Over baseline	0%	22.93%	-0.14%
k-img/s	91	~120	~120

New overall summary

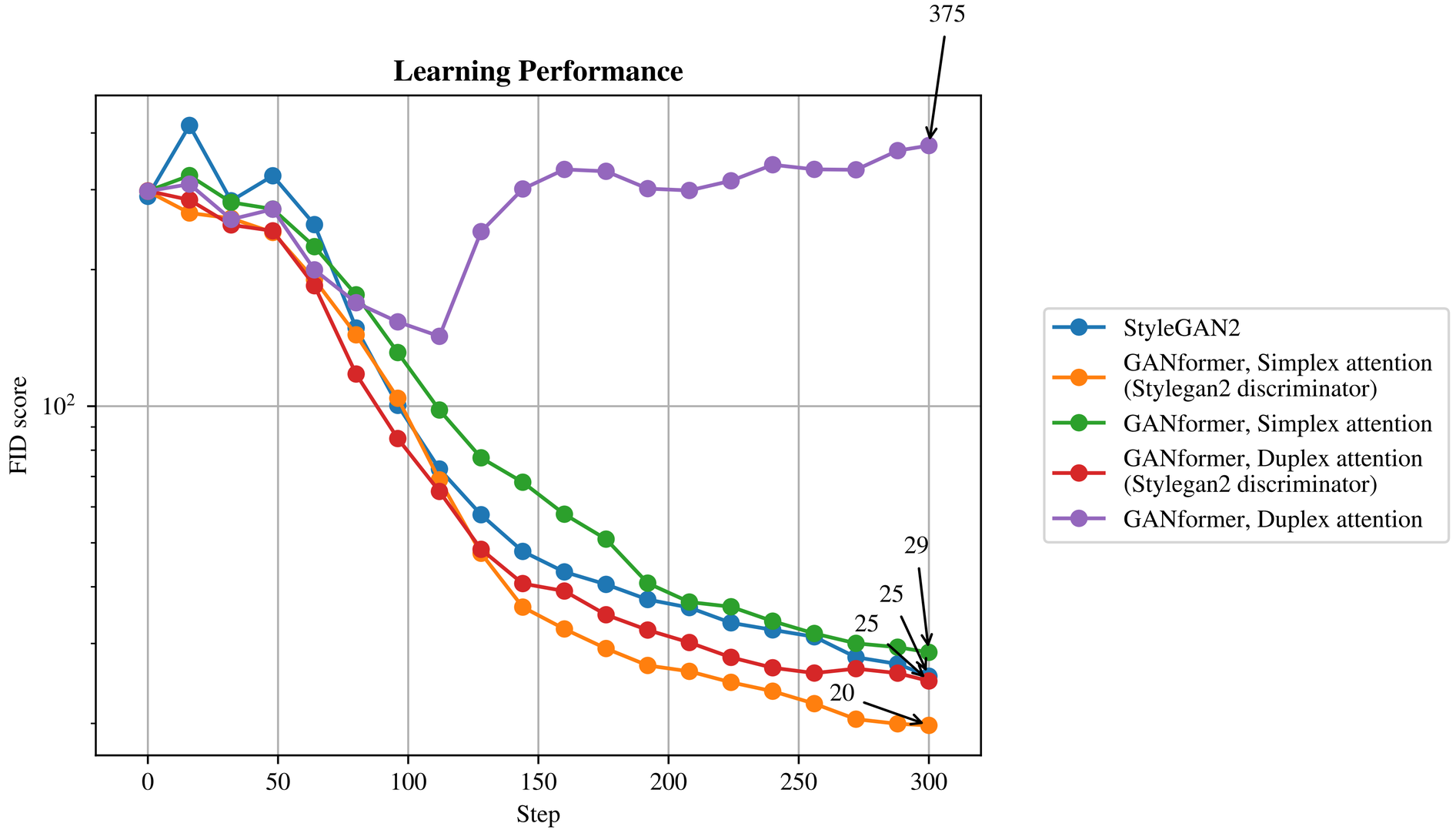
Paper's Results

Model	StyleGAN2	GANformer$_s$	GANformer$_d$
FID ↓	11.29	10.29	7.22
IS ↑	2.74	2.82	2.78
Precison ↑	52.02	56.76	55.45
Recall ↑	23.98	18.21	33.94
FID Improvement Over baseline	0%	8.86%	36.11%
k-img/s	Not given	Not given	Not given

New results

Model	StyleGAN2	GANformer$_s$ Stylegan2 Discriminator	GANformer$_d$ Stylegan2 Discriminator
FID ↓	24.77	19.09	24.81
IS ↑	2.50	2.62	2.62
Precison ↑	0.18	0.35	0.35
Recall ↑	2.11	4.76	2.11
FID Improvement Over baseline	0%	22.93%	-0.14%
k-img/s	91	~120	~120

Analysis of FID scores



Generated images

Dataset

StyleGAN2

GANformer
Simplex Attention /
StyleGAN2 Discriminator

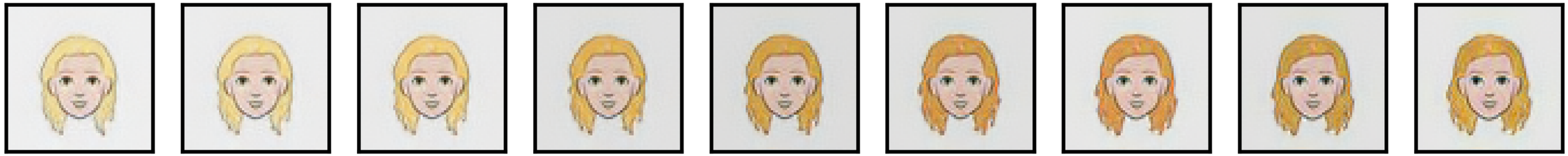
GANformer
Duplex Attention /
StyleGAN2 Discriminator

GANformer
Simplex Attention /
Simplex Attention
Discriminator

GANformer
Duplex Attention / Duplex
Attention Discriminator

Generated images: interpolation

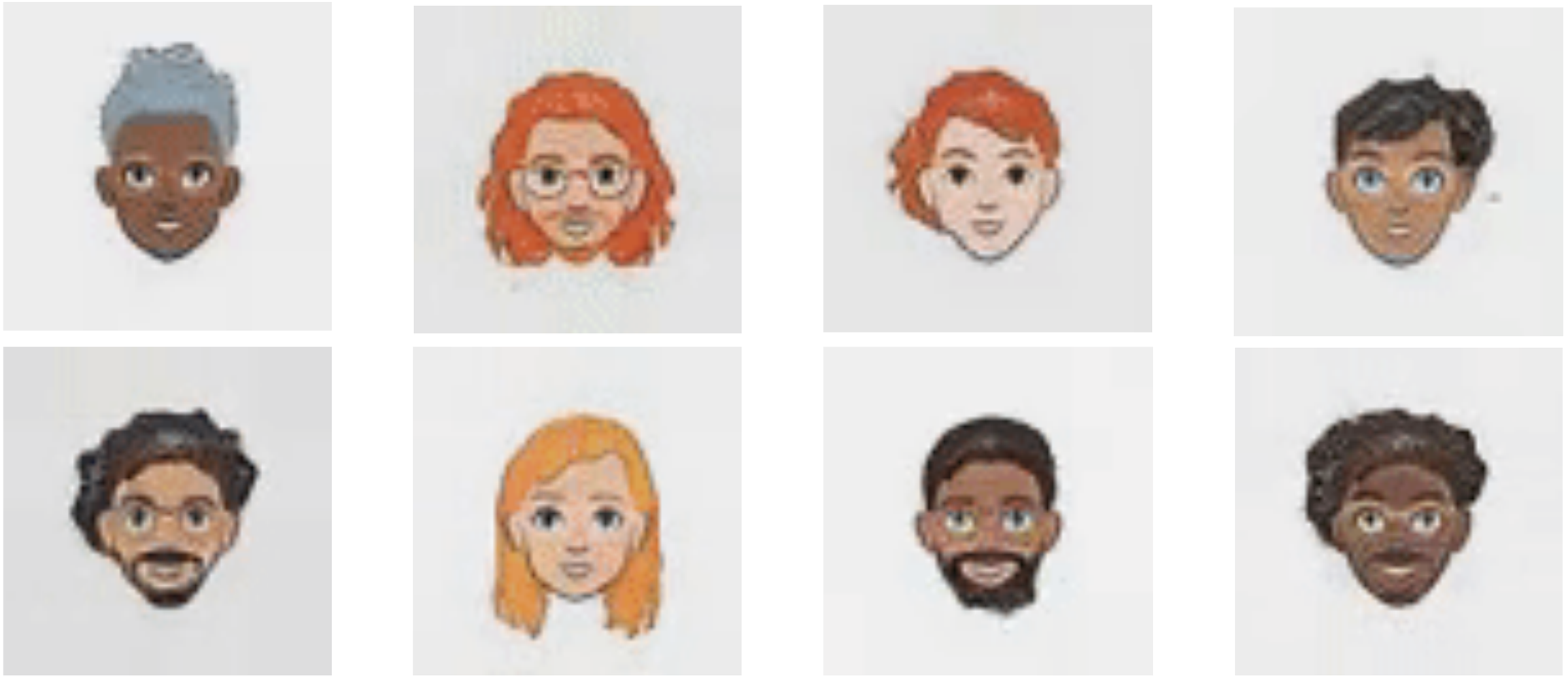
StyleGAN2



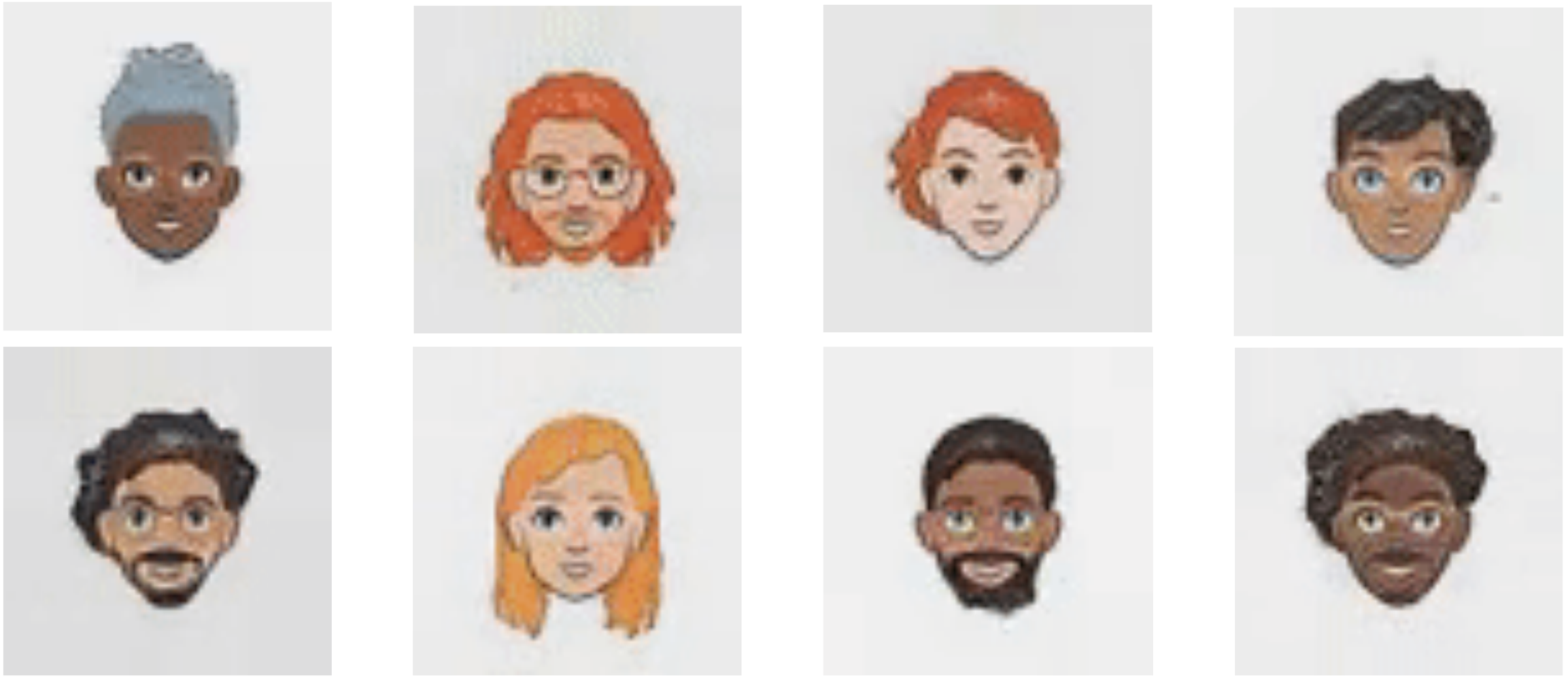
GANformer Simplex Attention with StyleGAN2 discriminator

GANformer Duplex Attention with StyleGAN2 discriminator

Generated images (best model)



Generated images (best model)



Style mixing

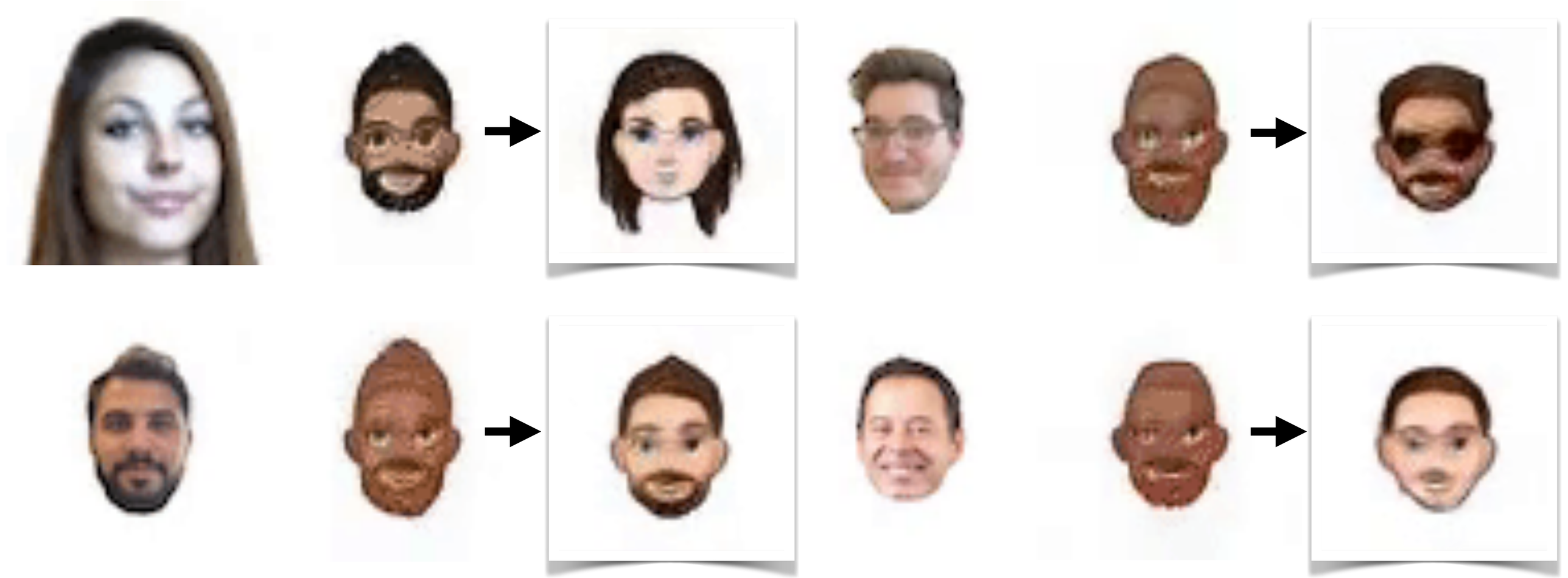
GANformer Simplex Attention with StyleGAN2 discriminator

Style mixing

GANformer Simplex Attention with StyleGAN2 discriminator

Style mixing

GANformer Simplex Attention with StyleGAN2 discriminator




CONCLUSIONS

Conclusion

- Reduced depth of our experiments:
  - ➡ Smaller and different dataset
  - ➡ Discard four out of the five baselines
  - ➡ Used models: StyleGAN2 and GANformer (Duplex and Simplex attention)
- Google Colab Pro compatible version of the authors code to run (54 hours total)
- Misleading paper claims and discrepancies w.r.t. code:
  - Discriminator attention
  - Hyperparameter discrepancies
  - Attention position on synthesis

➡ substituting the hypothetically valid discriminator network with a vanilla StyleGAN discriminator.

- New hybrid model performed significantly better than the baseline.
- Qualitative results
- Comment on results


The End

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14, page 2672–2680. MIT Press, 2014. URL http: //arxiv.org/abs/1406.2661.

[2] Drew A. Hudson and C. Lawrence Zitnick. Generative Adversarial Transformers, 2021. URL http://arxiv. org/abs/2103.01209.

[3] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 8110–8119, 2020. URL http://arxiv.org/abs/1912.04958.

[4] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 4401–4410, 2019. URL http://arxiv.org/abs/1812.04948.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008, 2017. URL http://arxiv.org/abs/1706.03762.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186. Association for Computational Linguistics, 2019. URL http://arxiv.org/abs/ 1810.04805.

[7] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2901–2910, 2017. URL http: //arxiv.org/abs/1612.06890.

References

[8] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop, 2015. URL http://arxiv.org/abs/ 1506.03365.

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3213–3223, 2016. URL http://arxiv.org/abs/1604.01685v2.

[10] Cole Forrester, Mosseri Inbar, Krishnan Dilip, Sarna Aaron, Maschinot Aaron, Freeman Bill, and Fuman Shiraz. Cartoon set. https://google.github.io/cartoonset/, 2018.

[11] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.

[12] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. Improved Training of Wasserstein GANs. arXiv:1704.00028 [cs, stat], December 2017. URL http://arxiv.org/abs/1704.00028. arXiv: 1704.00028 version: 3.

[13] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN. arXiv:1701.07875 [cs, stat], 2017. URL http://arxiv.org/abs/1701.07875.

[14] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization, 2017.