DAY 5: GENERATIVE MODELS, GENERATIVE ADVERSARIAL NETWORKS (GANS) BASICS

2021-05-07 | Mehdi Cherti | Cross Sectional Team Deep Learning, Helmholtz AI @ JSC

JÜLICH
Forschungszentrum

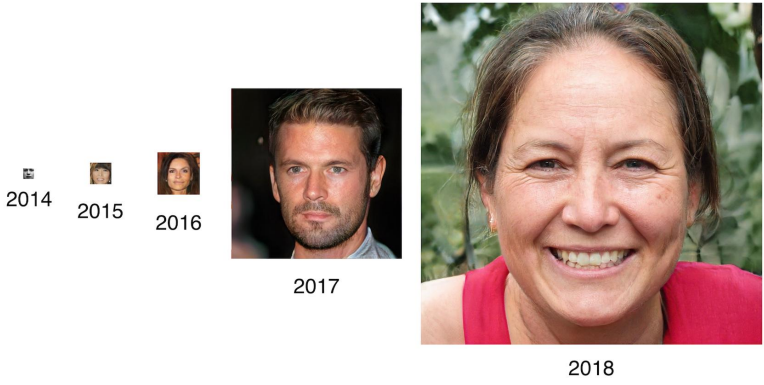
GENERATIVE MODELS

- Impressive progress in last years, algorithmic/architectural improvements coupled with large scale training
- Lot of different applications: image generation, text generation, speech synthesis, and more

Karras et al. (2020)

GENERATIVE MODELS

- Impressive progress in last years, algorithmic improvements coupled with large scale training
- Lot of different applications: image generation, text generation, speech synthesis, and more



(Source: `https://bit.ly/3azTV7J`)

GENERATIVE MODELS: BASICS

- The general setup: we have a set of samples $x_1, x_2, ..., x_N$ drawn i.i.d. from an unknown probability distribution $p(x)$. We would like to learn a model M, which we can use to generate samples from p
- Different formulations, training algorithms, architectures

Karras et al. (2020)

GENERATIVE MODELS, DENSITY MODELING

- We model the underlying data distribution of the data $p(x)$ with a surrogate distribution $q(x)$, i.e. $q(x) \approx p(x)$ for some similarity between probability distributions
- In generative modeling, x is in general a high-dimensional vector (e.g. image, sound, text)
- A common similarity measure used is the Kullback–Leibler (KL) divergence:

$$D_{\mathrm{KL}}(p \parallel q) = \int p(x) \log \frac{p(x)}{q(x)} dx$$

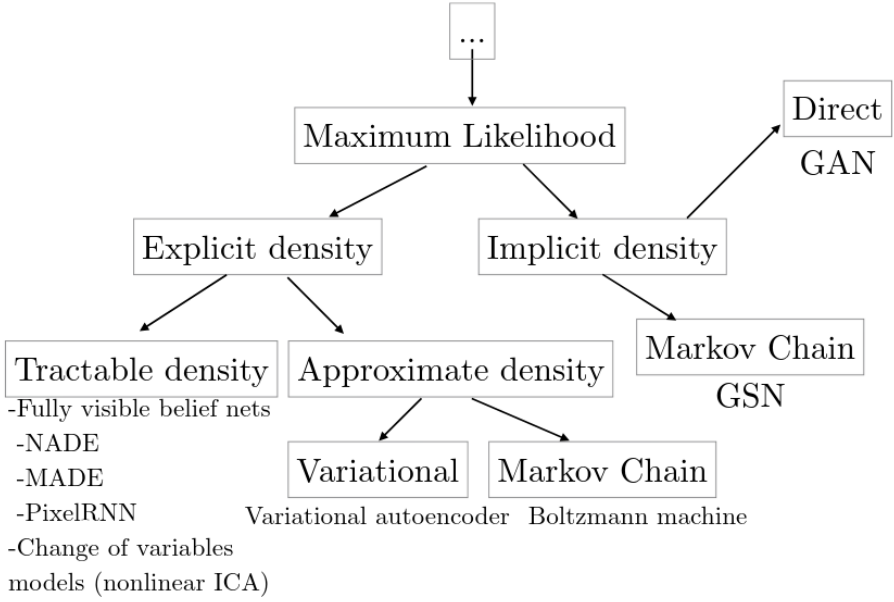
GENERATIVE MODELS, KL DIVERGENCE

- Connection to information theory and compression: $D_{\mathrm{KL}}(p \parallel q) = (-\int p(x) \log q(x) dx) - (-\int p(x) \log p(x) dx) = H(p, q) - H(p)$
- Connection to maximum likelihood: For a parametrized model q_Θ, maximization of the likelihood

$$\max_\Theta [E_{x \sim p} \log q_\Theta(x)] \approx \max_\Theta [\frac{1}{|D|} \sum_{x_i \in D} \log q_\Theta(x_i)]$$

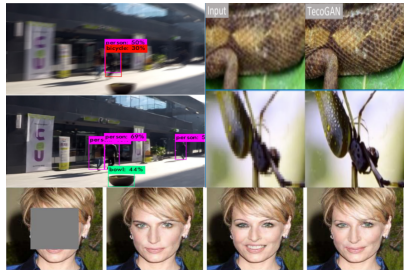
is equivalent to minimizing the KL divergence between q_Θ and p

TAXONOMY OF GENERATIVE MODELS



APPLICATIONS: IMAGE RECOVERY

- We can upscale images, denoise them, fill/recover missing parts (inpainting)
- In this case, we model $p(x|\tilde{x})$ where $\tilde{x}$ is x with some information destroyed ,e.g. by blurring or noise introduction, and we would like to recover that information
- Examples: TecoGAN (Chu et al. 2020), DeblurGAN (Kupyn et al. 2018), Palette (Saharia et al. 2021)



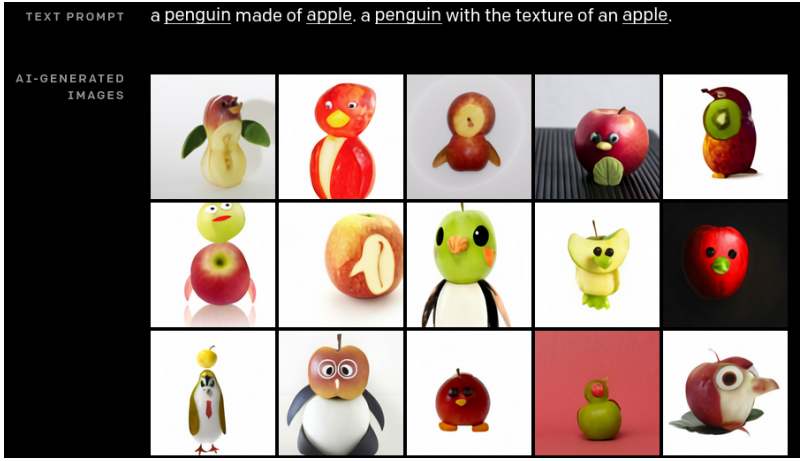
APPLICATIONS: CONDITIONAL IMAGE GENERATION

- We can generate images based on labels, or feature vectors, or natural language
- In this case, we model $p(x|y)$, where x is the image and y is the label, represented as a feature vector, a category, or a sequence of tokens ($y = y_1, ..., y_m$, where m is the number of tokens)

(Brock, Donahue, and Simonyan 2019)

APPLICATIONS: CONDITIONAL IMAGE GENERATION

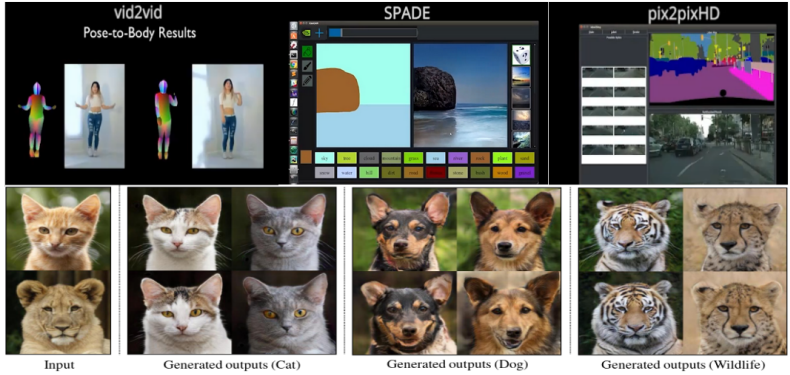
- What would a “penguin made of apples” look like?



OpenAI’s DALL-E `https://openai.com/blog/dall-e/`, also see GLIDE (Nichol et al. 2021)

APPLICATIONS: IMAGE-TO-IMAGE MODELS

- We can learn a mapping from images to images or from videos to videos, either with paired samples or unpaired ones. Examples: Pix2Pix (Isola et al. 2018), CycleGAN (Zhu et al. 2020), Pix2PixHD (Wang et al. 2018), SPADE (Park et al. 2019), StarGANv2 (Choi et al. 2020), Palette (Saharia et al. 2021)

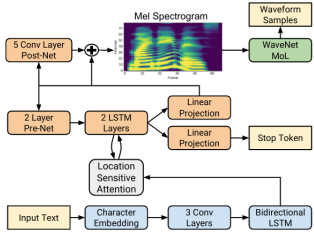


APPLICATIONS: SPEECH SYNTHESIS

- WaveNet

(Oord et al. 2016)

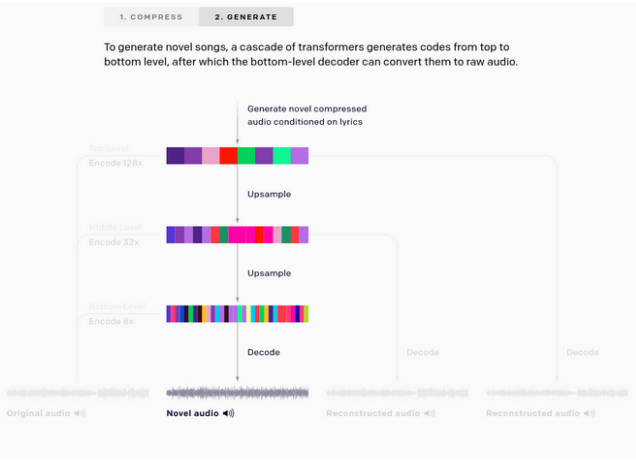
- Tacotron2



(Shen et al. 2018)

APPLICATIONS: MUSIC GENERATION

- Jukebox (`https://openai.com/blog/jukebox/`)



(Dhariwal et al. 2020)

APPLICATIONS: TEXT GENERATION

- GPT-3 (Brown et al. 2020)

```
[23] prompt = "Get salary details of the Workers whose AGE lies between 25 and 35"

[24] print(gpt.get_top_reply(prompt))

     output: Select Salary from Worker where AGE between 25 and 35;
```

Equation description

integral from a to b of f(t) with respect to t = F of b minus F of a

Translate

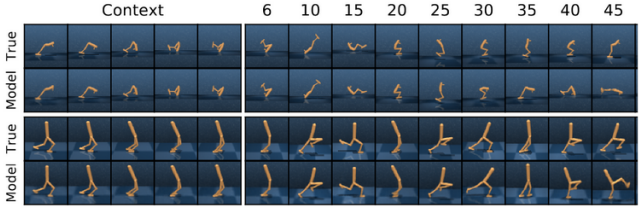
$$\int_a^b f(t)\,dt = \int_a^b \frac{F(b)-F(a)}{t}\,dt$$

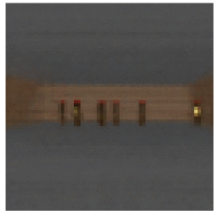
```
> python nlsh.py
nlsh> what day is it?
>>> Run:  date +%A [Y/n]:
Wednesday
nlsh> no, I mean the full date
>>> Run:  date +%F [Y/n]:
2020-06-10
nlsh> could you print them both together?
>>> Run:  date +%A %F [Y/n]:
date: illegal time format
usage: date [-jnRu] [-d dst] [-r seconds] [-t west] [-v[+|-]val[ymwdHMS]] ...
            [-f fmt date | [[[mm]dd]HH]MM[[cc]yy][.ss]] [+format]
nlsh> I think you forgot the quotes
>>> Run:  date '+%A %F' [Y/n]:
Wednesday 2020-06-10
nlsh> clone the openai gym repo and install it
>>> Run:  git clone https://github.com/openai/gym.git && cd gym && python setup.py install [Y/n]:
```

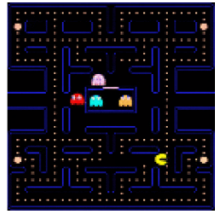
Source: https://bit.ly/3tqM8Sr, https://bit.ly/39KHNSe, https://bit.ly/3tvipHR

APPLICATIONS: REINFORCEMENT LEARNING

- We can simulate possible futures given the past. Application for reinforcement learning: learning world models and planning



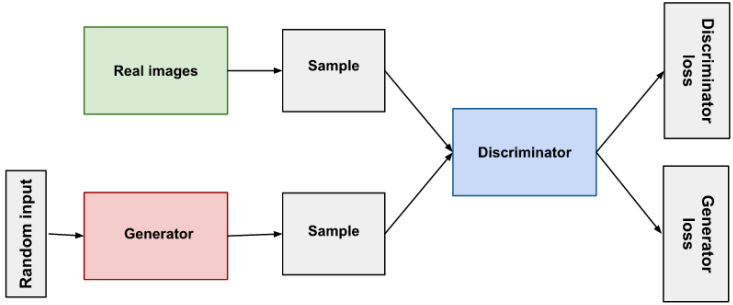




(Hafner et al. 2020), (Kim et al. 2020)

GENERATIVE ADVERSARIAL NETWORKS

- We have samples from a probability distribution p, and we would like to learn a generator model that generates samples from $x \sim p(x)$
- Discriminator is trained to distinguish between real and generated samples
- Generator is trained to fool the discriminator



GENERATIVE ADVERSARIAL NETWORKS – FORMULATION

$$\min_G \max_D \left[\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]\right]$$

- $D(x)$: probability that an image is real
- $D(G(z))$: probabilty that a generated image is real, where $z \sim \mathbb{N}(0, I)$
- $\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})]$: discriminator on real data
- $\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))]$: discriminator on generated data

GENERATIVE ADVERSARIAL NETWORKS – IDEAL CASE

- Generative Adversarial Networks (Goodfellow et al. 2014) shows that for a fixed generator G, the optimal discriminator is: $D^*_G(x) = \frac{p_{\text{data}}(x)}{p_{\text{data}} + p_g(x)}$
- They also show that the global optimal solution of the problem minimizes: $-\log 4 + 2 \cdot \text{JSD}(p_{\text{data}} \parallel p_g)$, where the JSD is a measure between probability distributions
- Since the JSD is non-negative, the globally optimal solution for the generator is the data distribution $p_g = p_{\text{data}}$

GENERATIVE ADVERSARIAL NETWORKS – TRAINING

- In practice, we alternate between updating the generator and updating the discriminator

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**

  for k steps **do**

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
  - Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  - Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  - Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

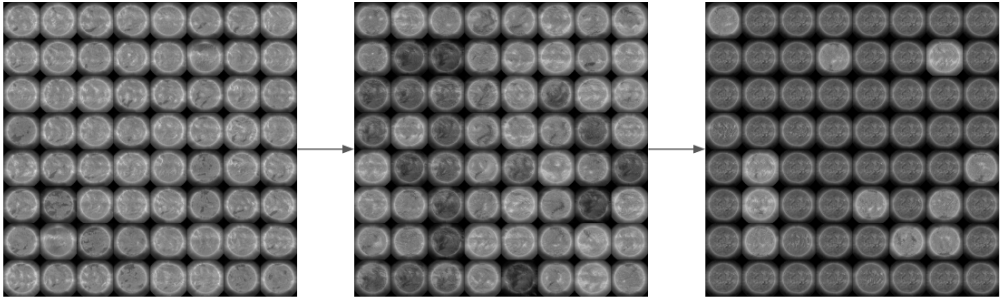
end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

- Note that in practice, for the generator they maximize $\log(D(G(\boldsymbol{z})))$ instead of minimizing $\log(1 - D(G(\boldsymbol{z})))$, because it provides better gradients

GENERATIVE ADVERSARIAL NETWORKS – ISSUES

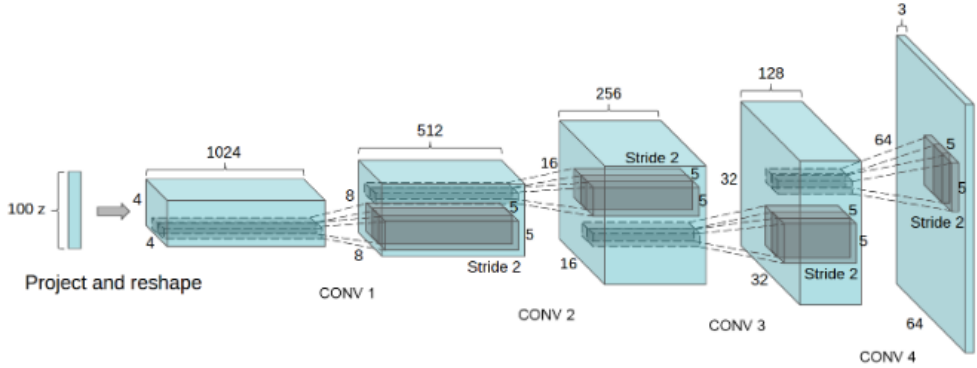
- Non-convergence
- Mode collapse
- Vanishing gradient



(Illustration of mode collapse)

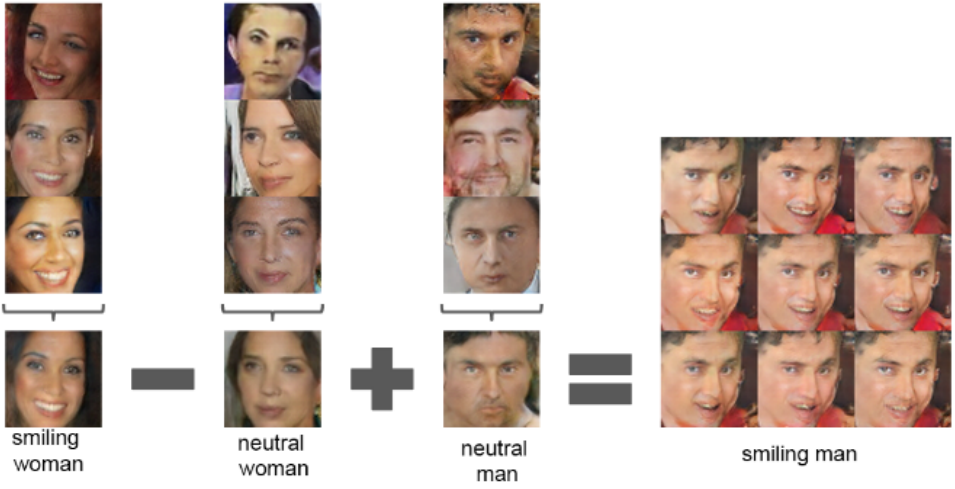
THE DCGAN ARCHITECTURE

- Removed fully connected layers: fully convolutional architecture for generator and discriminator
- Uses batch normalization to stabilize training
- One of the first architectures that worked well in practice on several datasets



THE DCGAN ARCHITECTURE – VECTOR ARITHMETICS

- Interpretable directions in the latent space



THE DCGAN ARCHITECTURE – INTERPOLATION IN LATENT SPACE

- Smooth interpolation between generated images using the latent space

EVALUATION METRICS

- In general, it's still an open question how to evaluate a generative model; in a lot of cases human visualization is still needed
- In the ideal case, metrics should be task-specific (Theis, Oord, and Bethge 2016) and evaluate your generative model depending on how you will use it
- Most common metrics used: Fréchet Inception Distance (FID) (Heusel et al. 2018), Inception Score (Salimans et al. 2016), precision and recall (Kynkäänniemi et al. 2019)

SUMMARY

- Impressive progress during last years
- Model sizes and data are getting bigger
- Lot of different applications: image generation, text generation, speech synthesis, and more

REFERENCES I

Brock, Andrew, Jeff Donahue, and Karen Simonyan. 2019. “Large Scale Gan Training for High Fidelity Natural Image Synthesis.” http://arxiv.org/abs/1809.11096.

Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, et al. 2020. “Language Models Are Few-Shot Learners.” http://arxiv.org/abs/2005.14165.

Choi, Yunjey, Youngjung Uh, Jaejun Yoo, and Jung-Woo Ha. 2020. “StarGAN V2: Diverse Image Synthesis for Multiple Domains.” http://arxiv.org/abs/1912.01865.

Chu, Mengyu, You Xie, Jonas Mayer, Laura Leal-Taixé, and Nils Thuerey. 2020. “Learning Temporal Coherence via Self-Supervision

REFERENCES II

for Gan-Based Video Generation.” ACM Transactions on Graphics 39 (4). https://doi.org/10.1145/3386569.3392457.
Dhariwal, Prafulla, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. 2020. “Jukebox: A Generative Model for Music.” http://arxiv.org/abs/2005.00341.
Goodfellow, Ian J., Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. “Generative Adversarial Networks.” http://arxiv.org/abs/1406.2661.
Hafner, Danijar, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. 2020. “Dream to Control: Learning Behaviors by Latent Imagination.” http://arxiv.org/abs/1912.01603.

REFERENCES III

Heusel, Martin, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2018. “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium.” http://arxiv.org/abs/1706.08500.

Isola, Phillip, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. 2018. “Image-to-Image Translation with Conditional Adversarial Networks.” http://arxiv.org/abs/1611.07004.

Karras, Tero, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. 2020. “Training Generative Adversarial Networks with Limited Data.” http://arxiv.org/abs/2006.06676.

REFERENCES IV

Kim, Seung Wook, Yuhao Zhou, Jonah Philion, Antonio Torralba, and Sanja Fidler. 2020. “Learning to Simulate Dynamic Environments with Gamegan.” http://arxiv.org/abs/2005.12126.

Kupyn, Orest, Volodymyr Budzan, Mykola Mykhailych, Dmytro Mishkin, and Jiri Matas. 2018. “DeblurGAN: Blind Motion Deblurring Using Conditional Adversarial Networks.” http://arxiv.org/abs/1711.07064.

Kynkäänniemi, Tuomas, Tero Karras, Samuli Laine, Jaakko Lehtinen, and Timo Aila. 2019. “Improved Precision and Recall Metric for Assessing Generative Models.” http://arxiv.org/abs/1904.06991.

REFERENCES V

Nichol, Alex, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. 2021. “Glide: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models.” arXiv Preprint arXiv:2112.10741.

Oord, Aaron van den, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. 2016. “WaveNet: A Generative Model for Raw Audio.” `http://arxiv.org/abs/1609.03499`.

Park, Taesung, Ming-Yu Liu, Ting-Chun Wang, and Jun-Yan Zhu. 2019. “Semantic Image Synthesis with Spatially-Adaptive Normalization.” `http://arxiv.org/abs/1903.07291`.

REFERENCES VI

Saharia, Chitwan, William Chan, Huiwen Chang, Chris A Lee, Jonathan Ho, Tim Salimans, David J Fleet, and Mohammad Norouzi. 2021. “Palette: Image-to-Image Diffusion Models.” arXiv Preprint arXiv:2111.05826.

Salimans, Tim, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. “Improved Techniques for Training Gans.” http://arxiv.org/abs/1606.03498.

Shen, Jonathan, Ruoming Pang, Ron J. Weiss, Mike Schuster, Navdeep Jaitly, Zongheng Yang, Zhifeng Chen, et al. 2018. “Natural Tts Synthesis by Conditioning Wavenet on Mel Spectrogram Predictions.” http://arxiv.org/abs/1712.05884.

REFERENCES VII

Theis, Lucas, Aäron van den Oord, and Matthias Bethge. 2016. “A Note on the Evaluation of Generative Models.” http://arxiv.org/abs/1511.01844.

Wang, Ting-Chun, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2018. “High-Resolution Image Synthesis and Semantic Manipulation with Conditional Gans.” http://arxiv.org/abs/1711.11585.

Zhu, Jun-Yan, Taesung Park, Phillip Isola, and Alexei A. Efros. 2020. “Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks.” http://arxiv.org/abs/1703.10593.