Increased Stability of GAN Training with Discriminator Stepback

Nicholas Larsen
Southern Methodist University
Dallas, TX
n.larsen@smu.edu

Steven Larsen
Southern Methodist University
Dallas, TX
s.larsen@smu.edu

***Abstract*—Training generative adversarial networks (GANs) is tediously difficult and can take massive amounts of time and resources. When using standard gradient descent techniques, the models will fail to converge because a step for either model changes the shape of the other's loss function. Often times training will result in volatile steps that could mean the next step is a worse result than the previous. We are proposing a new tool for training, by handicapping the discriminator, and giving the generator insight to where the discriminator might go next. We expect this to provide stability to training. This paper will be structured as followed: we will discuss related work in the field, the motivation behind our method, an algorithmic description of our method, the models used and some experiments to test our hypothesis.**

***Index Terms*—GAN Training, Deep Learning, Generative Adversarial Networks, Image Generation, Neural Networks**

I. Related Work

Generative adversarial networks (GANs) are an unsupervised learning technique [1] usually applied to generate images. Simply, there is two networks; first the generator, which generates data. A second network, the discriminator, is fed both images from the generator and real images. The discriminator attempts to identify the generated images. These networks are enabled to train from each other and remain largely unsupervised in the process. Their objective functions usually depend on each other's networks, thus resulting in a sporadic and unpredictable training process. The rest of this section will discuss recent improvements that have been made to GANs.

A. DCGAN

Raford et al. [1] wanted to use deep convolutional neural networks to succeed in the arena of unsupervised learning. The area they specifically targeted was GANs. Their method involved several alternations from previous GAN implementations. First, all pooling layers were replaced with strided convolutions and transpose convolutions in the discriminator and generator respectively. Next, numerous batch norm layers were added to both models. All generator layers used a rectified linear unit activation except the last convolution layer used a hyperbolic tangent activation. Lastly, the discriminator's activations were all leaky rectified linear units. This network was named the Deep Conversational Generative Adversial Network (DCGAN). Many of these techniques are used in the model below in section 3.

B. Least Square GANs

Li et al. [2] theorized that using cross entropy for the loss function in GANs creates a vanishing gradient problem. They created the network, Least Squares Generative Adversarial Network (LSGAN) using mean squared error that performed better than the previous Radford GAN. Li et al. introduced the technique of adding a class embedding to the fully connected layer in order to reduce mode collapse.

C. Training Improvements

Feature Matching was introduced by Salimans et al [3]. Feature Matching changes the generator's objective function to match statistics of the discriminator. Specifically, we will gather an intermediate layer's activation's in the discriminator and add the statistical differences to the objective function of the generator. For $f(x)$, a intermediate layer activation, an addition to the generator's objective function is as follow:[3]

$$||\mathbb{E}_{x\sim p_{data}} f(x) - \mathbb{E}_{x\sim p_z} f(G(z))||_2^2$$

Salimans et al[3]. suggest quite a few other improvements for GAN training, such as Minibatch discrimination, Historical averaging, One-sided label smoothing, and Virtual batch normalization. Lastly, they coined the term Inception Score, which we will be using to judge the quality of the distribution of images.

II. Motivation

We theorize that some of the volatility of GAN training can be attributed to the disparity in problem complexity of each model. In other words the Discriminator's task of distinguishing between real and fake images requires less training effort than generating images. A reasonable next progression is to step with the generator more often than the discriminator. Another goal would be to give the generator meaningful insight to what the discriminator might learn next. With these assumptions, we came up with **Discriminator weight set back**. Simply, every m epochs, replace the discriminator's weights with what they were n epochs ago. Theoretically, the generator has already seen what the discriminator could learn. The discriminator is disadvantaged during its learning process because it effectively steps less frequently.

III. Models

All of the following techniques listed in the DCGAN [1] were used:

- Replace all pooling layers with convolution and transpose convolutions.
- Use batch normalization in both models.
- Use a rectified linear activation unit for all the generator layers except the last layer, where we used a hyperbolic tangent function.
- Use a leaky rectified linear unit activation for the discriminator's activations.

A. CelebA Generator

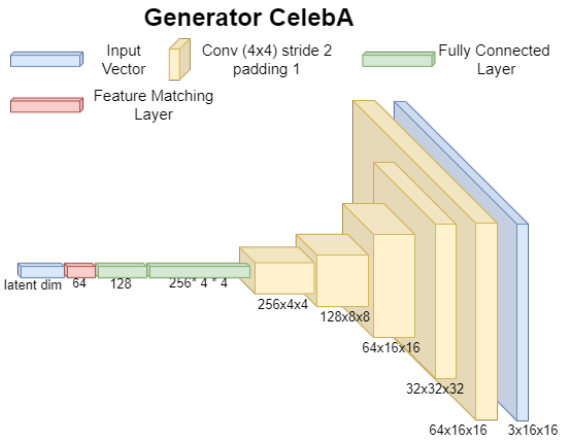

Fig. 1. Generator

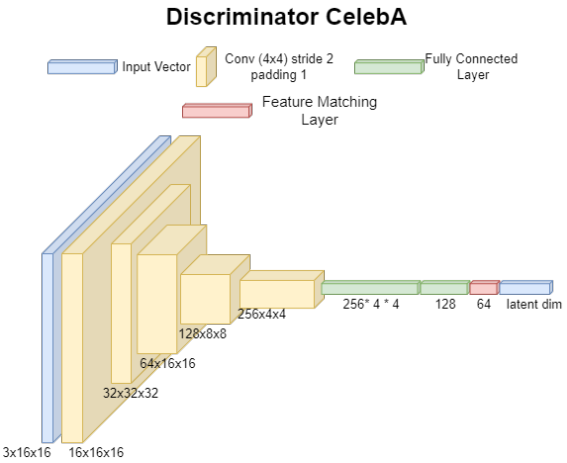

Fig. 2. Discriminator

IV. Algorithm

A. Traditional Methods

During training mean squared error was used as the loss function, inspied by Li et al.[2]. Feature matching was utilized to add to the loss of the generator Salimans et al[3]. In particular the output of the last fully connected layer was used as features. Mean squared error was use to compare the output of this layer for both real and generated images. Each network had a root mean squared propagation optimizer and the discriminator had twice the learning rate of the generator.

B. Discriminator Weight Setback

Our approach consists of selecting a SETBACK FREQUENCY and a SETBACK AMOUNT. During training both networks step every epoch. Once we have stepped SETBACK FREQUENCY times, the discriminator will be set to the state it was SETBACK EPOCHS ago.

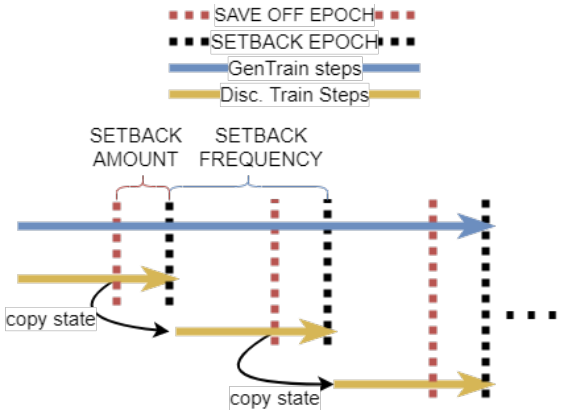

Fig. 3. Discriminator Setback

DISC ← DISCRIMINATOR
GEN ← GENERATOR
SAVED_DISC ← NULL
for EPOCH in EPOCHS **do**
  if EPOCH is a SAVEOFF EPOCH **then**
    SAVED_DISC ← DISC
  if EPOCH is a SETBACK EPOCH **then**
    DISC ← SAVED_DISC
  CALCULATE GEN LOSS
  STEP GEN
  CALCULATE DISC LOSS
  STEP DISC

Fig. 4. Discriminator Setback Alg.

V. Experiments

A. Experiment Description

For completeness, we ran a set of experiments with feature matching and without feature matching. Each set consisted of training a GAN on a subset of the aligned CelebA dataset in table 1.

setback frequency	setback percentage
Control	
100	25%
100	50%
100	75%
50	25%
50	50%
50	75%

TABLE I
EXPERIMENTS

B. Inception Score

Heavily inspired by the inception score described by Salimins et al[3], we developed a classifier that determined if a face is one of the following classes: **attractive** only, **mouth slightly open** only, **smiling** only, **attractive** and **mouth slightly open**, **attractive** and **smiling**, **mouth slightly open smiling**, **attractive mouth slightly open** and **smiling**, or **None** of those classes. We combined labels from the celebA dataset to get the aforementioned classes. This classifier will replace the inception model in the inception score calculation. These features were chosen with the goal of gathering a broad distribution of facial features in the dataset, however this can be improved upon to obtain a more sensitive scoring.

VI. RESULTS

A. Without Feature Matching

Inception Scores No Feature Matching		
Frequency	Percentage	Inception Score
100	25 %	5.45
100	50 %	5.11
100	75 %	5.87
50	25 %	5.02
50	50 %	5.56
50	75 %	6.10
Control		5.28

TABLE II

Table 2 contains inceptions scores of the models after training. This is an important measure that correlates with human preference [3]. Inspecting the inception score during training can give insight to the stability of the models.

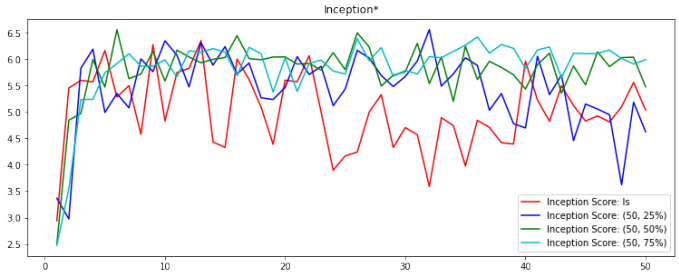

Fig. 5. Inception Score Over Time

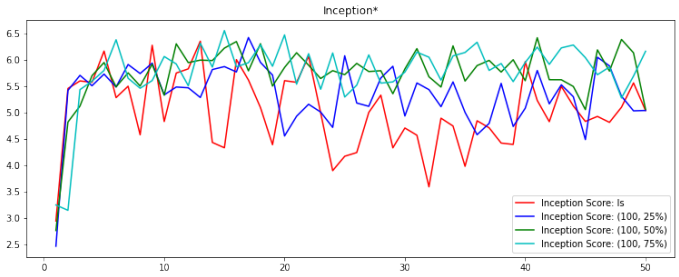

Fig. 6. Inception Score Over Time

When comparing the inception score over time there is a stability in our algorithm when compared to the control method, see figures 5 and 6. One trial is not conclusive, but the results below are images generated by these models. It is

Fig. 7. Examples Control without FM

worth noting there was a large amount of noticeable mode collapse around face 1 and 2 in the control examples. See appendix 1 for a vast amount of images from each generator

Fig. 8. Examples Frequency = 100 and Percentage = 25 without FM

Fig. 9. Examples Frequency = 100 and Percentage = 50 % without FM

In figure 8-13 there is more variety, and that is reflected in the Inception Score as well. A wide variety of these are available in the appendix for review. The results are not conclusive, however they suggest an improvement in performance of the generator.

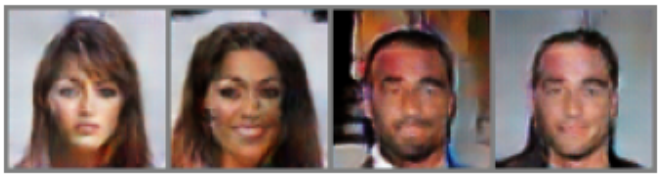

Fig. 10. Examples Frequency = 100 and Percentage = 75 % without FM

Fig. 11. Examples Frequency = 50 and Percentage = 25% without FM

Fig. 12. Examples Frequency = 50 and Percentage = 50 % without FM

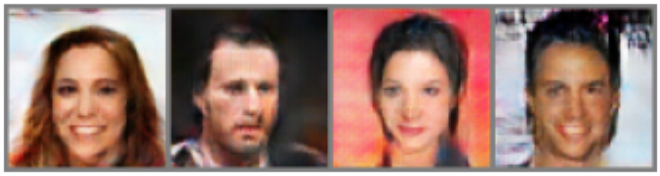

Fig. 13. Examples Frequency = 50 and Percentage = 75 % without FM

B. With Feature Matching

Figures 16-22 are the best examples of each model that was trained with feature matching. Table 3 shows the final inception scores after training. The model with frequency $=$ 100 and percentage $=$ 50% performed the best in terms of inception score and it also produced the best faces subjectively. When compared to the models without feature matching, as an aggregate, the feature matching models did worse. This may suggest that discriminator setback and feature matching are not compatible for training.

VII. Future Work

Some of the results in this paper suggest adding **discriminator setback** to training can improve image quality. Specifically considering inception score, models that were trained with **discriminator setback** performed better than the control with or without feature matching. Also, when plotting the inception score over epochs the models with **discriminator weight setback** appeared more stable with a few exceptions during feature matching training. To extend this research we recommend incorporating other methods suggested by Salimans et al. in *Improved Techniques for Training GANs*[3]. Evaluating this method heavily relies on the ability of the inception model,

Inception Scores Feature Matching		
Frequency	Percentage	Inception Score
100	25 %	5.56
100	50 %	5.92
100	75 %	5.92
50	25 %	5.50
50	50 %	5.81
50	75 %	5.91
Control		4.53

TABLE III

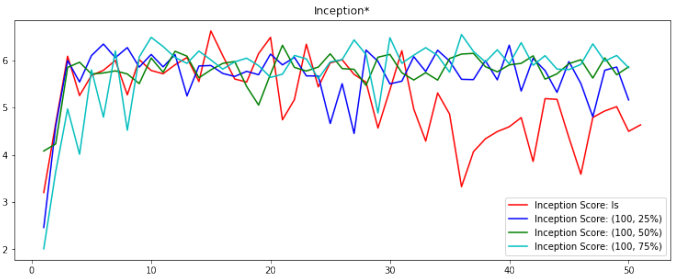

Fig. 14. Inception Score Over Time

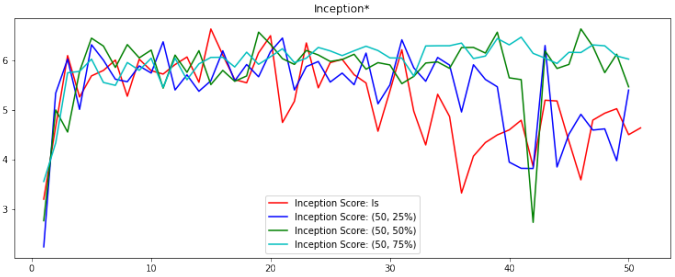

Fig. 15. Inception Score Over Time

Fig. 16. Examples Control with FM

Fig. 17. Examples Frequency = 100 and Percentage = 25 % with FM

Fig. 18. Examples Frequency = 100 and Percentage = 50 % with FM

Fig. 19. Examples Frequency = 100 and Percentage = 75 % with FM

Fig. 20. Examples Frequency = 50 and Percentage = 25 % with FM

Fig. 21. Examples Frequency = 50 and Percentage = 50 % with FM

Fig. 22. Examples Frequency = 50 and Percentage = 75 % with FM

used in the inception score, to predict a large distribution of the generated dataset. Improving the classifier to predict a wider range of attributes about faces would improve the meaningfulness of our inception score. They also suggest running inception score on $50k$ images, to provide more accurate results.

VIII. Appendix

Fig. 23. No Feature Matching Control

Fig. 24. No Feature Matching Setback = 50, Percentage = 25 %

Fig. 25. No Feature Matching Setback = 50, Percentage = 50 %

Fig. 26. No Feature Matching Setback = 50, Percentage = 75 %

Fig. 27. No Feature Matching Setback = 100, Percentage = 25 %

Fig. 28. No Feature Matching Setback = 100, Percentage = 50 %

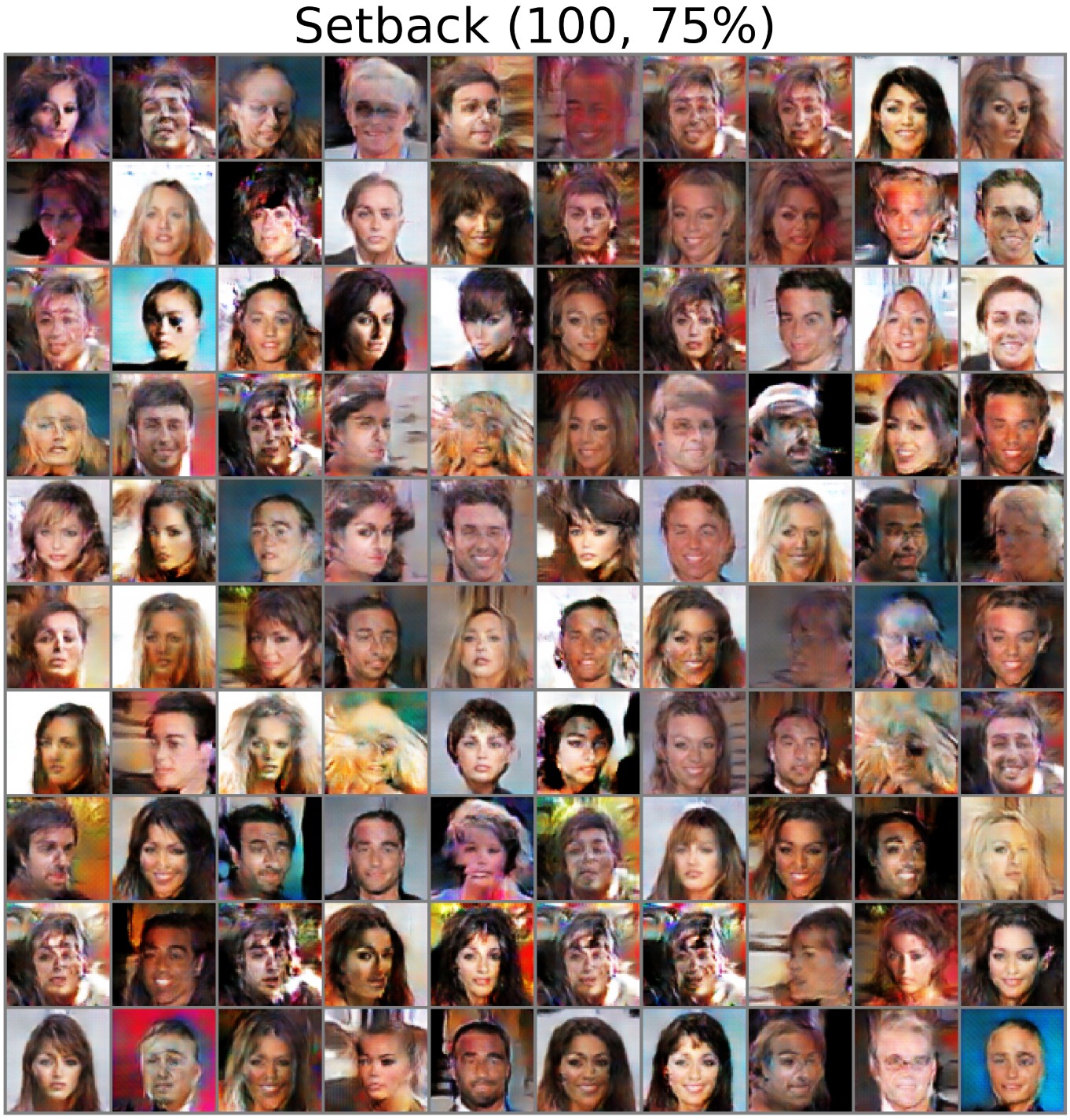

Fig. 29. No Feature Matching Setback = 100, Percentage = 75 %

Fig. 30. Feature Matching Control

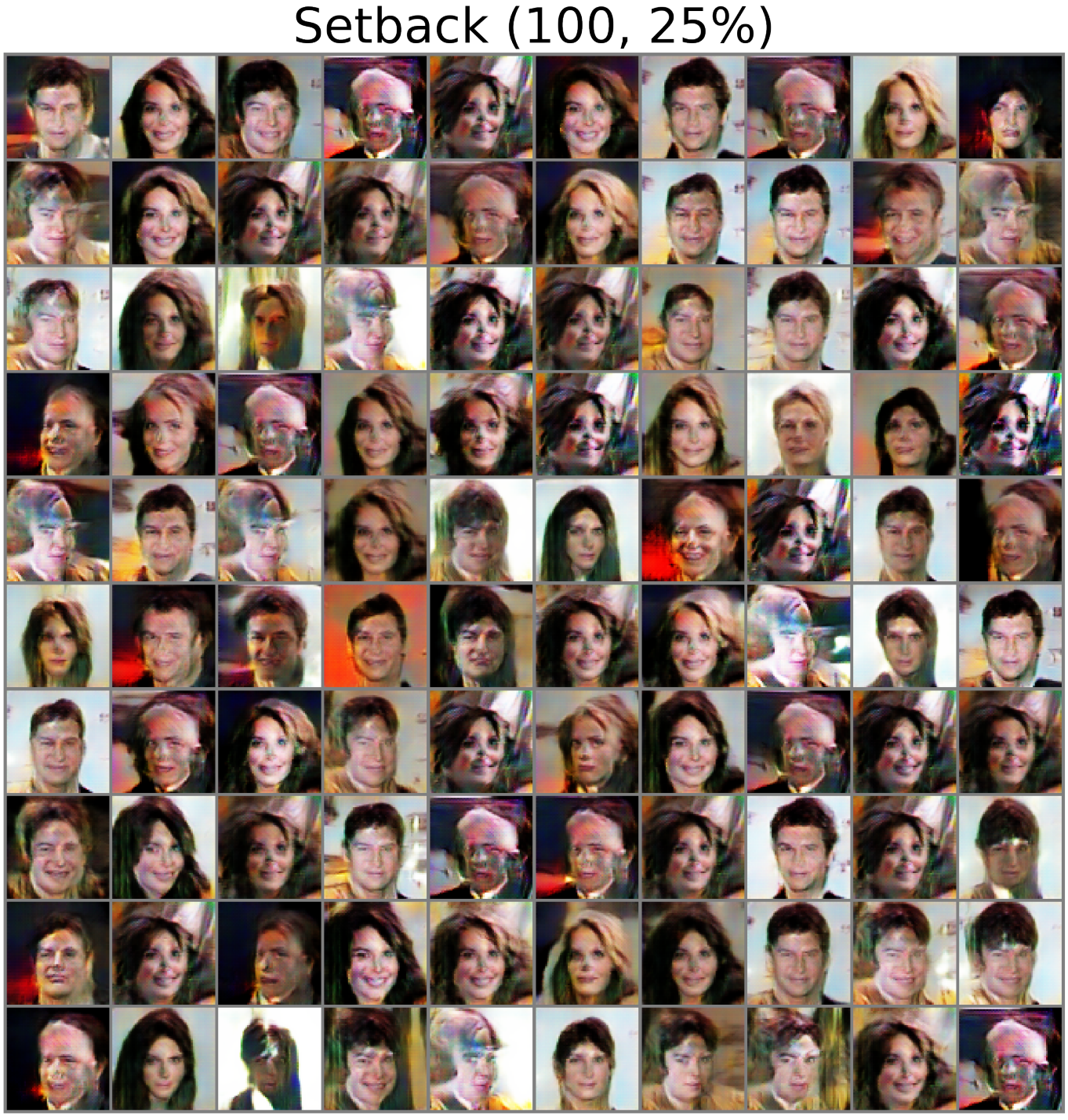

Fig. 31. Feature Matching Setback = 100, Percentage = 25 %

Fig. 32. Feature Matching Setback = 100, Percentage = 50 %

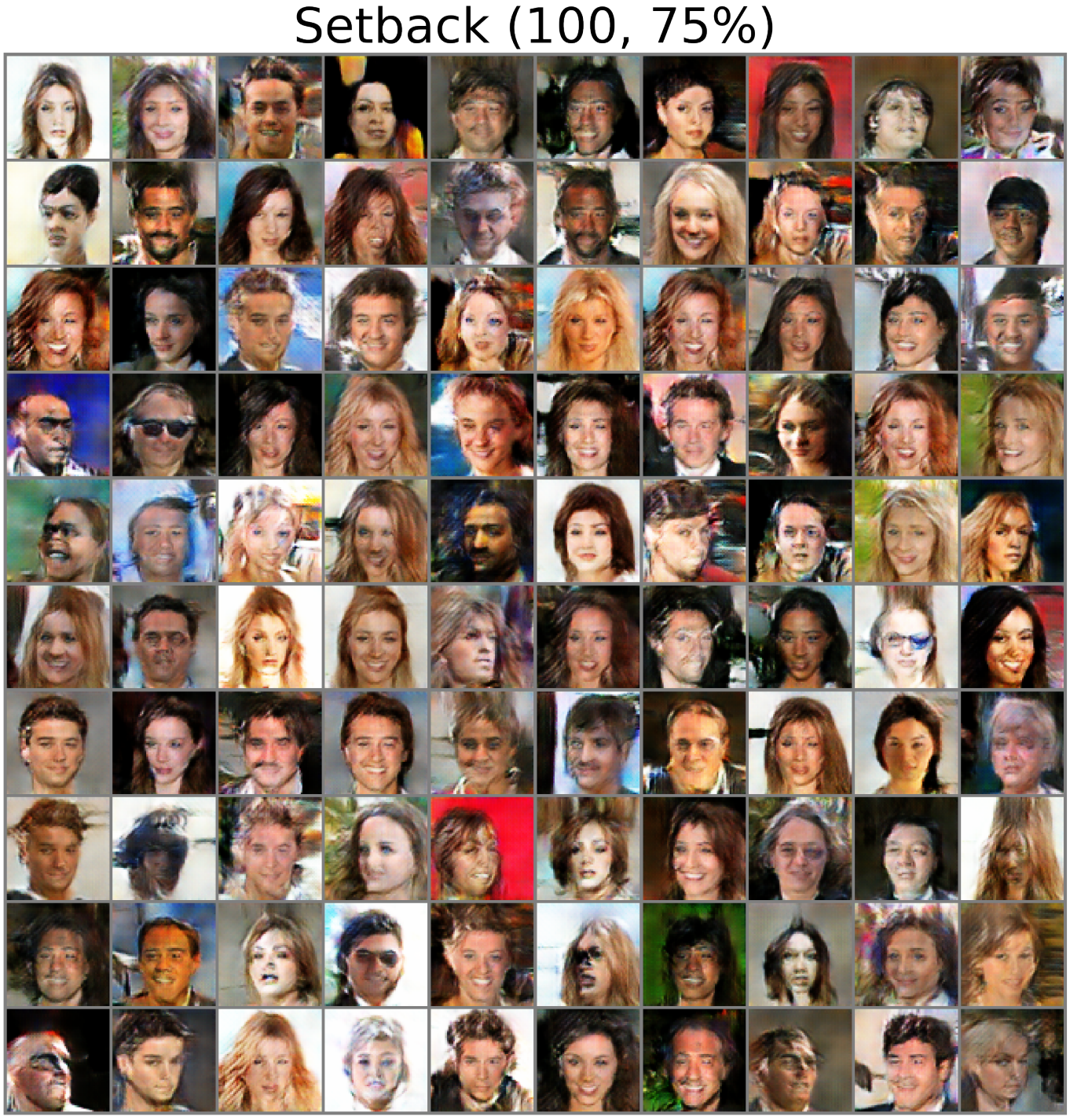

Fig. 33. Feature Matching Setback = 100, Percentage = 75 %

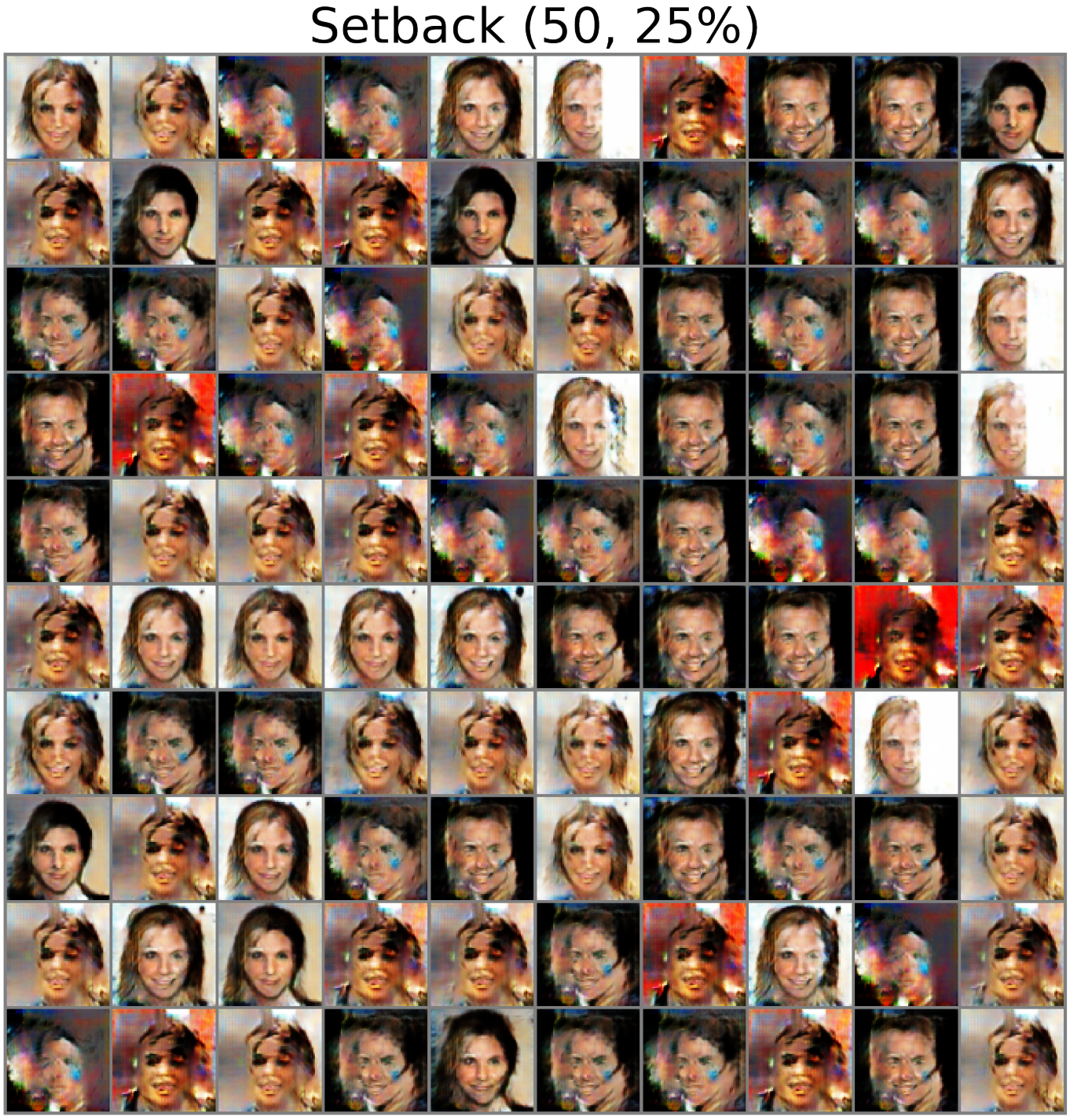

Fig. 34. Feature Matching Setback = 50, Percentage = 25 %

Fig. 35. Feature Matching Setback = 50, Percentage = 50 %

Fig. 36. Feature Matching Setback = 50, Percentage = 75 %

REFERENCES

[1] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in Advances in Neural Information Processing Systems (NIPS), pp. 2672–2680, 2014.

[2] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," in International Conference on Learning Representations (ICLR), 2015.

[3] Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. Improved techniques for training gans. arXiv:1606.03498, 2016

[4] Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096, 2018

[5] L. Mao, Y. Yan, J.-H. Xue, and H. Wang, "Deep multi-task multi-label CNN for effective facial attribute classification," IEEE Transactions on Affective Computing, pp. 1–1, 2020.

[6] M. Arjovsky and L. Bottou, "Towards Principled Methods for Training Generative Adversarial Networks." arXiv, 2017. doi: 10.48550/ARXIV.1701.04862.