Efficient CIFAR-100 Modelling

Faiza Jalil
nfqn37

Abstract

This paper adapts Residual Network (ResNet) architecture using depthwise separable convolutions for efficient CIFAR-100 classification and presents a modified conditional Spectral Normalization Generative Adversarial Network (SN-GAN) for image generation, also from CIFAR-100. The adapted SN-GAN features adaptive weighting in hinge loss calculations and utilizes conditional batch normalization and a projection discriminator to ensure stable training and targeted image synthesis.

Part 1: Classification

1 Methodology

Our methodology enhances the ResNet architecture by integrating depthwise separable convolutions, aiming to optimize parameter efficiency. ResNets, leveraging skip connections, enable training of deeper networks by learning residual functions relative to the input layers, effectively addressing the vanishing gradient problem [1]. We adopt depthwise separable convolutions, a strategy introduced by MobileNetV1 [2], replacing standard convolutions to diminish the number of operations and, consequently, runtime, while maintaining performance parity. This technique comprises depthwise convolutions, which apply a unique filter to each input channel, followed by pointwise convolutions that aggregate these outputs using 1×1 convolutions. Both layers incorporate batch normalization and ReLU nonlinearities. The depthwise convolution operation, denoted as $\hat{G}_{k,l,m}$, for an input feature map F, with a kernel $\hat{K}$ of size $D_K \times D_K \times M$, is defined as:

$$\hat{G}_{k,l,m} = \sum_{i,j} \hat{K}_{i,j,m} \cdot F_{k+i-1,l+j-1,m}, \tag{1}$$

where the mth filter in $\hat{K}$ is applied to the mth channel in F, generating the mth channel of the output feature map $\hat{G}$. The computational cost for depthwise separable convolutions is expressed as:

$$D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F, \tag{2}$$

significantly reducing the computational load compared to standard convolutions. Our adapted ResNet architecture employs a specific configuration of layers, beginning with an initial convolutional layer of 32 channels, followed by four main residual blocks with 32, 64, 96, and 128 channels respectively, each comprising depthwise separable convolutions for efficient feature processing. The network concludes with an adaptive average pooling layer followed by a fully connected layer, designed to output 100 classes.

To elevate our model's performance, we incorporated data augmentation strategies such as random cropping, flipping, color jittering, and normalization to reduce overfitting risk. Considering the limited number of optimization steps, we utilized a larger batch size of 512 to expose the model to more data in fewer iterations. Hyperparameter tuning revealed a learning rate of 0.001 to be most effective. While weight decay was considered for regularization, it was not adopted due to its negative impact on convergence given our step constraints.

2 Results

Our classification model, comprising 94,372 parameters, underwent 10,000 training iterations. Upon completion, it achieved a training accuracy of $65.5\% \pm 2.6\%$ and a test accuracy of $57.7\% \pm 2.3\%$, indicating

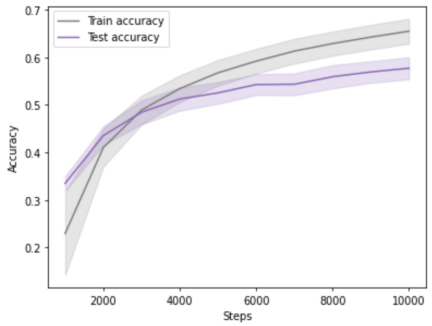

Figure 1: Training and Test Accuracy during Training

a generalization gap of 7.8%. This discrepancy suggests a tendency towards overfitting. Nonetheless, the model's low variance across iterations signifies stability in its performance.

Figure 1 illustrates that the model's accuracy plateaus beyond 6000 steps, albeit with slight improvements, implying potential for further generalisability with extended training. Empirical tests conducted by doubling the channel sizes, resulting in 334,052 parameters, have demonstrated a notable enhancement in performance (65% test accuracy), reinforcing the hypothesis that a larger model capacity could yield improved results.

3 Limitations

The achieved test accuracy of 57% suggests that the model is capturing underlying patterns in the data, but the 7.9% disparity between training and test accuracy indicates a propensity for overfitting. Enhancing the model with more parameters and extended training periods elevate its feature extraction proficiency but may further exacerbate overfitting concerns. To counteract this, implementation of stronger regularization strategies or architectural alterations, including the use of inverted residuals and linear bottlenecks as introduced in MobileNetV2 [6], could be beneficial.

Part 2: Generative model

4 Methodology

GANs comprise two fundamental elements: a discriminator D, which assesses whether samples are drawn from the actual data distribution $x \sim p_{\text{data}}$, and a generator G, which generates samples from latent variables $z \sim p_z$ in an attempt to replicate p_{data}. Through adversarial training, D aims to maximize its accuracy in identifying real versus generated samples, while G endeavors to produce samples that D classifies as real. This interaction is formalized as a mini-max game optimizing the value function $V(G, D)$:

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim p_{\text{data}}}[\log D(x)] + \mathbb{E}_{z \sim p_z}[\log(1 - D(G(z)))]. \tag{3}$$

In contrast, our chosen architecture for the generative task is based on the *aw-SN-GAN* model as introduced in [7], wherein we employ a hinge loss function instead to facilitate adversarial training. Hinge loss is a margin-based loss function that promotes the generation of realistic samples by penalizing discriminator predictions that are incorrect with confidence beyond a certain margin. It has demonstrated performance improvements for SN-GANS [5] and the formalized loss for the discriminator and generator is as follows:

$$L_D^{\text{hinge}} = \mathbb{E}_{x \sim p_{\text{data}}}\left[\min(0, -1 + D(x))\right] + \mathbb{E}_{z \sim p(z)}\left[\min(0, -1 - D(G(z)))\right], \tag{4}$$

$$L_G^{\text{hinge}} = -\mathbb{E}_{z \sim p(z)}\left[D(G(z))\right]. \tag{5}$$

The architecture adopts an adaptive weighting scheme to modify the discriminator's hinge loss function L_{aw}^D as delineated in [7].

$$L_{\text{aw}}^D = w_r \cdot L_r + w_f \cdot L_f, \tag{6}$$

with

$$L_r = \mathbb{E}_{x \sim p_{\text{data}}} \left[\max(0, 1 - D(x)) \right],$$
$$L_f = \mathbb{E}_{z \sim p(z)} \left[\max(0, 1 + D(G(z))) \right],$$

where w_r and w_f represent dynamically adjusted weights for the real (L_r) and fake (L_f) losses, respectively. Utilizing gradients ∇L_r and ∇L_f, we steer ∇L_{aw}^D as per the paper's algorithm. This approach mitigates negative impacts between losses, enhancing stability, and reducing mode collapse risk. We adopt the official code and recommended parameters from [7].

In our *aw-SN-GAN* model adaptation, Spectral Normalization is employed on the discriminator's convolutional layers, enhancing training stability and sampling quality. The model employs the spectral norm $SN(A) = \max_{\|h\|_2 \leq 1} \|Ah\|_2$ to normalize weights W, setting $W_{SN} = \frac{W}{SN(W)}$ for 1-Lipschitz continuity. This method uses power iteration, preferring it over singular value decomposition for efficiency. Starting with random vectors $v \in \mathbb{R}^n$ and $u \in \mathbb{R}^m$, we perform iterative updates $u_{t+1} = Wv_t$ and $v_{t+1} = W^T u_{t+1}$ to approximate $SN(W)$ with $u^T W v$. Due to slight weight variations during optimization, a single iteration each step suffices, keeping v and u optimally aligned.

We also integrate conditional generation into the *aw-SN-GAN* model using the provided class labels, following the projection mechanism detailed in [4]. This approach allows for the efficient integration of conditional information into the discriminator without the substantial parameter increase seen with traditional vector concatenation methods. The projection-based method is formalized as:

$$f(x, y) := y^\top V \phi(x) + \psi(\phi(x)), \tag{7}$$

where $y^\top$ denotes the transposed one-hot class label vector, facilitating direct interaction with the discriminator's feature vector $\phi(x)$ extracted from the input x. The matrix V serves to embed class vectors for sophisticated label mapping, while $\psi(\phi(x))$ performs a transformation on $\phi(x)$, enhancing the discriminator's ability to verify the authenticity and relevance of the input to its corresponding class label. Moreover, the generator employs conditional batch normalization (CBN), adjusting normalization parameters for each class label to aid the generation of class-distinctive images.

Model Architecture

Table 1 presents the generator and discriminator architectures, adapted for parameter efficiency, summarizing the integration of key methods previosuly discussed.

Table 1: The architecture of the adapted *aw-SN-GAN* model.

Generator	**Discriminator**
Input: $z \in \mathbb{R}^{128}$	Input: RGB image $x \in \mathbb{R}^{32 \times 32 \times 3}$
ConvTranspose2D up 128	SN Conv2D down 3
CBN + LeakyReLU	SN Conv2D down 16
ConvTranspose2D up 64	SN Conv2D down 32
CBN + LeakyReLU	SN Conv2D down 32
ConvTranspose2D up 64	SN Conv2D down 64
CBN + LeakyReLU	SN Conv2D down 64
ConvTranspose2D up 32	SN Conv2D down 128
CBN + LeakyReLU	FC to Scalar
Output: Tanh	Output: FC + Projection

5 Results

The final model, featuring 942,812 parameters, underwent training over 50,000 iterations on the full CIFAR-100 dataset, achieving an FID score of 38.27 against the CIFAR-100 test set. In comparison, a non-conditional model variant recorded a slightly improved FID of 32.00. Nevertheless, the conditional model was preferred for its ability to generate class-specific images, enable interpretable interpolations by facilitating easier sampling for high-quality outputs.

Figure 2 showcases a randomly sampled batch from the model's latent space, paired with random class labels. The images exhibit clarity, realistic textures with some shapes appropriate to their labels, demonstrating the model's ability to capture a diverse array of classes with minimal mode collapse.

Figure 3 illustrates linear interpolations among high-fidelity images generated for diverse classes, namely *poppy*, *apple*, *mountain*, *cloud*, *rocket*, *cattle*, *willow tree*, and *whale*. These interpolations not only retain realism but also ensure recognizability at their midpoints, evidencing the model's adeptness in transitioning smoothly across the dataset's varied distributions.

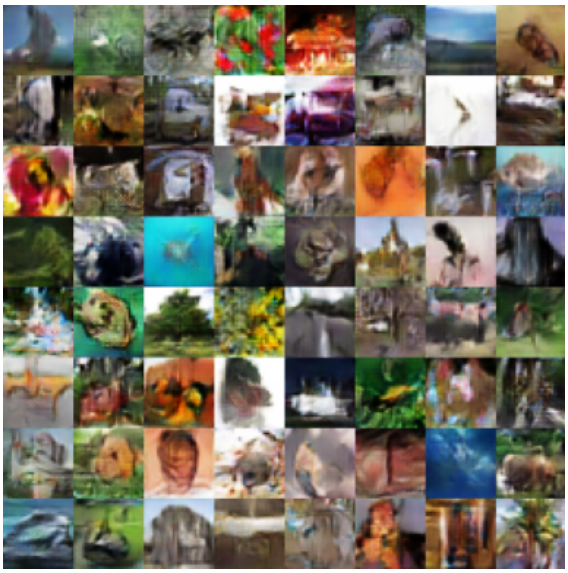

Figure 2: A random batch of 64 non-cherry picked samples.

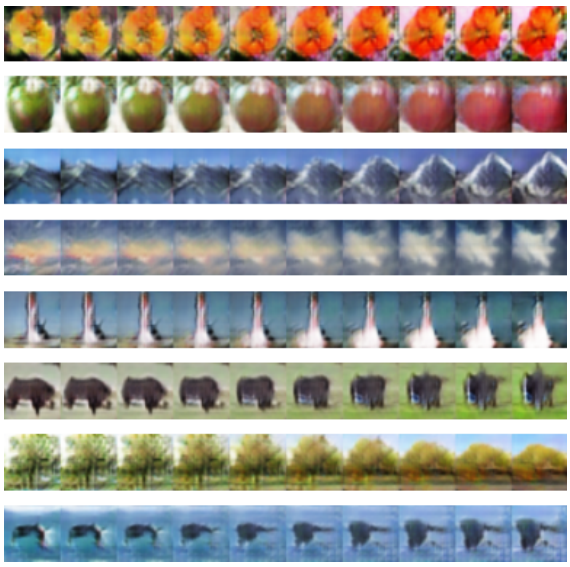

Figure 3: Interpolations within the latent space across various classes.

Employing the LPIPS metric yielded a score of 0.23. This result signifies a moderate level of perceptual dissimilarity, suggesting that the generated images exhibit some distinct features when compared to their nearest neighbors in the training dataset.

The interpolations underscore the model's adeptness in rendering simpler entities, such as *trees* and *apples*, whereas it encounters difficulties with more intricate subjects, like vehicles and human figures. Nevertheless, distinct successful instances as shown in Figure 4, particularly the depictions of a *pickup truck* and a *woman*, highlight the model's capacity to represent complex categories and morphologies, evidencing its potential for diverse and intricate image generation.

Figure 4: Cherry-picked examples illustrating successful class representations.

6 Limitations

Our GAN model occasionally generates undistinguishable instances within a sampled batch of 64 images and exhibits a degree of nondistinctiveness relative to the training dataset. However, despite limitations in parameters, training time, and data resolution, the model achieves outputs with realistic shapes, colours and textures relevant to their classes. Adopting more efficient architectures like Depthwise Convolutions or Skip Excitation layers as proposed in [3] could enable faster training efficiency and model capacity, potentially improving FID scores.

References

[1] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: 2015. arXiv: 1512.03385 [cs.CV].

[2] Andrew G. Howard et al. "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications". In: 2017. arXiv: 1704.04861 [cs.CV].

[3] Bingchen Liu et al. *Towards Faster and Stabilized GAN Training for High-fidelity Few-shot Image Synthesis*. 2021. arXiv: 2101.04775 [cs.CV].

[4] Takeru Miyato and Masanori Koyama. "cGANs with Projection Discriminator". In: 2018. arXiv: 1802.05637 [cs.LG].

[5] Takeru Miyato et al. "Spectral Normalization for Generative Adversarial Networks". In: *International Conference on Learning Representations*. 2018. URL: https://openreview.net/forum?id=B1QRgziT-.

[6] Mark Sandler et al. *MobileNetV2: Inverted Residuals and Linear Bottlenecks*. 2019. arXiv: 1801.04381 [cs.CV].

[7] V. Zadorozhnyy, Q. Cheng, and Q. Ye. "Adaptive Weighted Discriminator for Training Generative Adversarial Networks". In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 2021, pp. 4779–4788. DOI: 10.1109/CVPR46437.2021.00475. URL: https://doi.org/10.1109/cvpr46437.2021.00475.