

The Impact of Quantisation on Performance of Generative Adversarial Networks

Luka Lukač, Damjan Strnad

University of Maribor

Faculty of Electrical Engineering and Computer Science

Koroška cesta 46, 2000 Maribor, Slovenia

{luka.lukac, damjan.strnad}@um.si

Abstract. *This paper focuses on analyzing the influence of quantisation techniques on performance of generative adversarial networks (GANs) to reduce computational and memory requirements during their deployment. Two common techniques, post-training quantisation (PTQ) and quantisation-aware training (QAT), are incorporated into the adversarial training in multiple precision formats. Experimental results on two datasets from different modalities show that quantised models significantly reduced network and memory load with acceptable reduction of performance. The success of quantisation, however, strongly depends on the choice of quantisation strategy and targeted models.*

Keywords. machine learning, neural network, GAN, quantisation-aware training, post-training quantisation

1 Introduction

Modern generative adversarial networks (GANs) have exhibited remarkable results in sample generation across various domains (Aftabi et al., 2023; Majidpour et al., 2026; P. Wang and Sertel, 2023). Their performance, however, comes at the cost of requiring substantial computational resources and memory (Lin et al., 2021). Different compression schemes have been developed for conventional neural network (NN) architectures to address these challenges, including model pruning (Y. He et al., 2019), low-rank decomposition (Li et al., 2023), knowledge distillation (Qin et al., 2021) and quantisation (Rokh et al., 2023). Among these, quantisation stands out as a particularly effective and widely adopted approach for reducing model size and memory load. However, despite its frequent occurrence and success in other machine learning schemes, its application to GANs remains limited.

The main idea behind quantisation is to represent NN parameters with fewer bits than with standard 32-bit floating-point precision (Yang et al., 2019). This reduction leads to smaller model size and faster inference, enabling deployment on resource-constrained devices. However, naïve quantisation typically re-

sults in significant performance degradation (Y. Guo, 2018). Several more sophisticated approaches such as post-training quantisation (PTQ) and quantisation-aware training (QAT) (Hasan, 2024) were proposed to mitigate this issue. Despite the emergence of many effective quantisation methods, they have been rarely studied in GAN training (P. Wang et al., 2019), largely due to notorious instability and difficulty of adversarial training (Wiatrak et al., 2019). Therefore, this study investigates the impact of different quantisation strategies on performance of GANs.

The main contributions of this paper are:

- A study of quantisation techniques and their application on GANs with respect to their specifics for an efficient quantisation procedure.
- An experimental evaluation of PTQ and QAT approaches that demonstrate their effectiveness in quantising GAN models.

The remainder of the paper consists of 4 sections: Section 2 provides the background behind GANs and quantisation techniques, Section 3 describes the application of quantisation on GANs, Section 4 depicts and explains the results of the experimental work while Section 5 concludes the paper and provides cues for future work.

2 Background

This section consists of two parts. In the first part, the basic principles behind model quantisation are explained. After that, in the second part, GANs are shortly reviewed.

2.1 Quantisation

The fundamental idea behind quantisation is to represent a high-precision NN model (typically 32-bit floating point (FP32)) with lower-precision data types such as 16-bit floating point (FP16), 16-bit brain floating point (BF16), 8-bit integer (INT8) or 4-bit integer (INT4). Among these, BF16 is the most recent data type developed by Google Brain (S. Wang and Kanwar, 2019) and has the identical dynamic range to FP32 but

lower precision. Quantisation methods can be split into two main groups (Wei et al., 2024):

- Post-training quantisation (PTQ) where models are quantised after the training phase.
- Quantisation-aware training (QAT) that integrates the quantisation process into model training or fine-tuning.

2.1.1 Post-Training Quantisation

The less complex among the two is PTQ where a model is trained in high precision and is quantised only after the model has been trained to the full extent (Nagel et al., 2021). In the case of conversion to an integer precision data type, an original floating-point value r is rescaled to a quantised integer value q as:

$$q = \text{round}(r/S + Z) \quad (1)$$

where S represents a positive downscaling factor while Z is a shifting zero-point value. After rounding, q is often clamped to the respective integer-type range to prevent overflow.

S and Z are usually determined during a calibration process to ensure maximum quantisation efficiency. To be able to effectively map floating-point values to a limited integer precision, statistics of the activations distributions are gathered. Consequently, the downside of PTQ is the need for a representative calibration dataset, with which the calibration is performed.

There exist several common PTQ algorithms for calculating S and Z (ApX Machine Learning, 2025). The most simple is MinMax quantisation where the minimum and maximum values of a quantised tensor are used directly to calculate S and Z . Despite the simple implementation, MinMax is highly sensitive to outliers. Percentile quantisation technique is often used as a more robust alternative, where values below the lower percentile threshold and above the upper percentile threshold (e.g., 0.1% and 99.9%) are clamped. After that, the threshold values are used in calculation of the quantisation parameters. Percentile quantisation scheme is therefore much more resistant to outliers but introduces saturation error for the clipped values. The final major algorithm is entropy quantisation, where clipping thresholds are iteratively determined via Kullback-Leibler (KL) divergence. This approach often results in best performance of the quantised model but is computationally much more expensive than MinMax and percentile quantisation.

2.1.2 Quantisation-Aware Training

PTQ can, in some cases, significantly degrade the performance of a quantised model (Hubara et al., 2021). Consequently, QAT, which integrates quantisation into the training or fine-tuning phase, was proposed. The full-precision model learns to adapt its parameters during the training in a way that the performance of the

quantised model does not plunge. Consequently, the negative impact of QAT can be significantly reduced in comparison with PTQ.

Execution of the entire training process in low-precision arithmetic is unviable due to gradient calculation instability (Micikevicius et al., 2017). Therefore, QAT inserts special layers into the model computation graph during training, which quantise tensors and immediately dequantise them, simulating operations in lower-bit precision. The problem occurs in backward pass, as rounding and flooring operations, occurring in quantisation function, often have zero gradients. To solve this issue, the straight-through estimator (STE) technique is used (Bengio et al., 2013), which bypasses the problematic operations and calculates gradients only on truly differential steps in the training pipeline.

2.2 Generative Adversarial Networks

GANs are popular models that have been extensively used for sample generation purposes (Aggarwal et al., 2021; Creswell et al., 2018; Gui et al., 2023). A vanilla GAN consists of two NNs: the first is a generator, which produces samples from a randomly sampled latent space that resemble specimens from the original data distribution, and the second is a discriminator, which is trained to qualify samples as either real or artificially generated. The both models are trained alternately to the extent where the discriminator cannot distinguish between real and fake samples. A schematic overview of a GAN is shown in Fig.1.

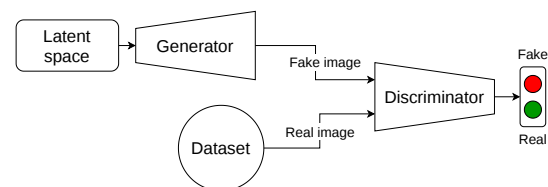


Figure 1. A basic GAN scheme.

Mode collapse is one of the most frequent issues appearing in GAN training (Kossale et al., 2022). Several techniques have approached this challenge in different ways in the past (Metz et al., 2016; Nguyen et al., 2017; Salimans et al., 2016). However, a real breakthrough in tackling mode collapse was the introduction of Wasserstein GANs (WGANs) (Arjovsky et al., 2017). By using the Wasserstein distance instead of divergence losses, the stability of the training was greatly improved while mode collapse was much rarely encountered. Several modern GAN architectures emerged after WGAN, including BigGAN (Brock et al., 2018) and StyleGAN (Karras et al., 2018) that were successfully applied to different fields (Z. Guo et al., 2026; Vanhala and Kurvinen, 2026; X.-Y. Zhang et al., 2025).

3 Method

This section contains three parts. Firstly, the GAN training is explained briefly. After that, in the following two parts, the methodology behind PTQ and QAT is described.

3.1 Training of a GAN

Let D be a discriminator and let G be a generator of a GAN, which is trained for observing quantisation influence on its performance. To enable control over generated samples, a conditional GAN (cGAN) (Mirza and Osindero, 2014) is employed. Discriminator loss function $\mathcal{L}_D$ is based on a modern hinge loss (Kavalerov et al., 2021) and is structured as:

$$\mathcal{L}_D = \mathbb{E}_{(x,y) \sim \mathbb{P}_r} [\max(0, 1 - D(x, y))] + \mathbb{E}_{z \sim \mathcal{N}(0, I), y \sim \mathbb{P}_r} [\max(0, 1 + D(G(z|y)|y))] \quad (2)$$

where x represents a real data sample from the data distribution $\mathbb{P}_r$, z represents a latent vector that is sampled from the standard Gaussian distribution $\mathcal{N}(0, I)$, y represents a class label and $\sim$ represents a sampling operator. To impose 1-Lipschitz constraint inside the critic, spectral normalisation is used (Z. Zhang et al., 2020).

On the other hand, the generator loss function $\mathcal{L}_G$ is defined as:

$$\mathcal{L}_G = -\mathbb{E}_{z \sim \mathcal{N}(0, I), y \sim \mathbb{P}_r} [D(G(z|y)|y)] + \lambda_{div} \mathcal{L}_{div} \quad (3)$$

where λ_{div} is a scaling factor and $\mathcal{L}_{div}$ is a diversity term defined as:

$$\mathcal{L}_{div} = -\mathbb{E} \left[\frac{\|G(z_1, y) - G(z_2, y)\|_1}{\|z_1 - z_2\|_1 + \epsilon} \right] \quad (4)$$

where z_1 and z_2 are different latent vectors and ϵ is a term that prevents dividing by 0.

Algorithmically, the GAN training loop is presented in Algorithm 1. The training is executed in a classical way for N iterations (Line 7), where D and G are trained in an adversarial fashion. In Lines 10–12, D is trained with real and generated data while in Lines 14–16, G is trained with generated data only. The result of the training are the trained G and D in Line 18.

Algorithm 1 Training loop of the GAN.

```

1: function GAN-TRAIN( $G, D, N, B, \eta_G, \eta_C$ )
2:      $\triangleright N$ : number of iterations
3:      $\triangleright B$ : batch size
4:      $\triangleright \eta_G, \eta_C$ : generator and critic learning rate
5:      $\triangleright \lambda_{div}$ : diversity loss factor
6:
7:     for  $N$  iterations do
8:          $x, y \leftarrow \text{LoadRealSamples}(B)$ 
9:
10:         $z \leftarrow \text{SampleLatentVectors}(B)$ 
11:         $\mathcal{L}_D \leftarrow \text{DiscLoss}(x, z, y)$   $\triangleright$  Eq. 2
12:         $D \leftarrow \text{Backpropagation}(\mathcal{L}_D, \eta_C)$ 
13:
14:         $z \leftarrow \text{SampleLatentVectors}(B)$ 
15:         $\mathcal{L}_G \leftarrow \text{GeneratorLoss}(z, y, \lambda_{div})$   $\triangleright$  Eq. 3
16:         $G \leftarrow \text{Backpropagation}(\mathcal{L}_G, \eta_G)$ 
17:    end for
18:    return  $G, D$ 
19: end function

```

3.2 Post-Training Quantisation

In the PTQ approach, both G and D are trained in full floating-point precision. After the training is complete, only G is quantised as there are no use cases for D in the inference phase. In the case of quantisation to integer-type format, more sensitive activations are calibrated with entropy quantisation while weights are calibrated per channel with a MinMax approach. The advantage in GAN quantisation is that there is no need for a specific calibration dataset as additional samples can be easily generated with G .

3.3 Quantisation-Aware Training

During QAT, G and D are also trained in full precision. However, fake quantisation modules are inserted to one or both NNs. Tensors are then quantised and immediately dequantised during a forward pass, simulating a realistic quantisation error of lower-precision data types. After the training is done, G is converted to the target precision and is used for inference purposes.

4 Results

The main goal of the experiments was to analyse the impact of quantisation on performance of GANs. Two datasets from different modalities were used: a vector dataset HWCD (Lukač, 2026) and a raster image dataset CIFAR-10 (Krizhevsky et al., 2009). The experiments were conducted on a personal computer with Intel i9-12900K CPU, NVIDIA RTX 3080 GPU and 64 GB of RAM on Windows 11 Education OS.

Both G and D were designed as conditional residual networks (K. He et al., 2016) where G progressively increases sample resolution while on the other

hand D progressively decreases it. The HWCD generator consisted of 4 residual blocks and an output 1D convolutional layer with linear upsampling between them while its discriminator consisted of three fully-connected layers. On the other hand, the CIFAR-10 generator consisted of three residual blocks, self-attention block and an output 2D convolutional head. The CIFAR-10 discriminator was designed with three downsampling residual blocks and a linear output layer. Adam optimizer was used in all experimental settings with betas 0.0 and 0.9. The training hyperparameters are collected in Table 1.

Table 1. Training hyperparameters.

	HWCD	CIFAR-10
Number of iterations	100,000	
Batch size	512	128
Latent vector size	100	128
Generator learning rate	0.002	0.001
Discriminator learning rate	0.002	0.001
Label embedding size	32	
Diversity term factor	0.5	

Results of quantisation on the vector dataset HWCD are collected in Table 2. FP32 represents the baseline model while FP16, BF16 and INT8 represent quantised models. QAT was performed on G only, D only and on both G and D , while PTQ was applied exclusively to G . Although quantising D alone in QAT has no practical use in the inference phase, these experiments were performed for completeness purposes. A pre-trained classifier trained on a subset of HWCD was used for performance estimation on the generated samples with a trained G . It can be seen that the FP32 baseline yielded the best results, but quantisation to FP16 and BF16 introduced slight deterioration in classification accuracy. When quantising to INT8, the classification accuracy was on-par if applying quantisation to D only but it plunged more when quantising G or both G and D . Furthermore, QAT proved to be superior to PTQ in most of the test scenarios, which can be attributed to the training adapted to quantified activations and weights.

Table 2. Classification accuracy of generated HWCD samples in regards to different quantisation and casting setups.

	QAT			PTQ
	G	D	$G+D$	G
FP32	97.93%			
FP16	97.88%	97.66%	96.56%	97.07%
BF16	97.74%	97.51%	97.81%	96.93%
INT8	93.63%	97.89%	88.76%	2.55%

Table 3 collects the quantisation results on the

standard benchmark raster image dataset CIFAR-10. Unlike at HWCD, the quantisation performance was measured with the Fréchet inception distance (FID) (Heusel et al., 2017), which indicates better results in the case of lower values. As expected, FP32 yielded the best results. Larger differences between quantisation approaches emerged: the results were significantly better when only D is quantised than when G or both G and D underwent the QAT. Quantised G to INT8 resulted in unstable training, the result of which was modest performance. When quantising to FP16 or BF16, PTQ appeared to be superior to QAT, mainly due to the more stable training in FP32.

Table 3. FID between real and generated CIFAR-10 samples in regards to different quantisation and casting setups.

	QAT			PTQ
	G	D	$G+D$	G
FP32	19.55			
FP16	29.71	22.27	26.66	20.05
BF16	29.54	22.11	26.73	20.31
INT8	81.32	23.10	121.15	360.89

Table 4 summarises the G size under different numerical precisions for the HWCD and CIFAR-10 datasets. Reducing the numerical precision leads to decreased model sizes (50% for FP16 and BF16, and 75% for INT8). This indicates that quantisation can significantly shrink the G size during the inference phase without modifying the model architecture. Careful numerical precision selection, however, is crucial to preserve good model performance while reducing memory load.

Table 4. Memory usage of the quantized generator.

	HWCD	CIFAR-10
FP32	12.40 MB	16.53 MB
FP16, BF16	6.20 MB	8.26 MB
INT8	3.10 MB	4.13 MB

Due to lower memory usage and calculations in lower precision, the inference speed is higher and energy consumption is lower. However, it is difficult to directly measure those two effects as they significantly vary across different GPU architectures and underlying operating systems. Therefore, they are not evaluated directly in this work.

5 Conclusion

This paper presents the incorporation of quantisation to GAN models and analyses its impact on their performance. Two common quantisation techniques were

used: PTQ where one or both models were quantised in the post-training phase, and QAT where quantisation was applied already during the training. Both approaches were examined on a vector dataset HWCD and a raster image dataset CIFAR-10. The experiments indicate that quantisation can be highly beneficial for memory requirements and inference speed. However, the quantisation strategy must be selected carefully to obtain negligible deterioration in model performance.

In the future, the performance of quantised models should be tested in practical applications. Deriving from this, a thorough analysis of energy savings would be highly beneficial to observe the trade-off between computational efficiency and performance of GANs. Furthermore, it would be beneficial if the impact of quantisation were observed on other datasets from different data modalities.

Acknowledgments

The authors acknowledge the financial support from the Slovenian Research and Innovation Agency as a part of Young Researcher Funding under Grant 0796-59772, and Research Core Funding P2-0041.

References

- Aftabi, S. Z., Ahmadi, A., & Farzi, S. (2023). Fraud detection in financial statements using data mining and GAN models. *Expert Syst. Appl.*, 227, 120144. <https://doi.org/10.1016/j.eswa.2023.120144>
- Aggarwal, A., Mittal, M., & Battineni, G. (2021). Generative adversarial network: An overview of theory and applications. *Int. J. Inf. Manag. Data Insights*, 1(1), 100004. <https://doi.org/10.1016/j.jjimei.2020.100004>
- ApX Machine Learning. (2025). Common PTQ algorithms. Retrieved May 7, 2026, from <https://apxml.com/courses/practical-llm-quantization/chapter-2-post-training-quantization-ptq/common-ptq-algorithms>
- Arjovsky, M., Chintala, S., & Bottou, L. (2017, January). Wasserstein GAN. <https://doi.org/10.48550/arXiv.1701.07875>
- Bengio, Y., Léonard, N., & Courville, A. (2013). Estimating or propagating gradients through stochastic neurons for conditional computation. <https://doi.org/10.48550/arXiv.1308.3432>
- Brock, A., Donahue, J., & Simonyan, K. (2018). Large scale GAN training for high fidelity natural image synthesis [Version Number: 2]. <https://doi.org/10.48550/arXiv.1809.11096>
- Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B., & Bharath, A. A. (2018). Generative adversarial networks: An overview. *IEEE Signal Process. Mag.*, 35(1), 53–65. <https://doi.org/10.1109/MSP.2017.2765202>
- Gui, J., Sun, Z., Wen, Y., Tao, D., & Ye, J. (2023). A review on generative adversarial networks: Algorithms, theory, and applications. *IEEE Trans. Knowl. Data Eng.*, 35(4), 3313–3332. <https://doi.org/10.1109/TKDE.2021.3130191>
- Guo, Y. (2018). A survey on methods and theories of quantized neural networks. <https://doi.org/10.48550/arXiv.1808.04752>
- Guo, Z., Du, W., Li, C., & Liu, Z. (2026). Generative adversarial networks in health management of rotating machinery: Survey, recent advances and perspectives. *Neurocomputing*, 681, 133390. <https://doi.org/10.1016/j.neucom.2026.133390>
- Hasan, J. (2024). Optimizing large language models through quantization: A comparative analysis of PTQ and QAT techniques. <https://doi.org/10.48550/ARXIV.2411.06084>
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. <https://doi.org/10.1109/CVPR.2016.90>
- He, Y., Liu, P., Wang, Z., Hu, Z., & Yang, Y. (2019). Filter pruning via geometric median for deep convolutional neural networks acceleration. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). GANs trained by a two time-scale update rule converge to a local Nash equilibrium. *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems*, 6629–6640.
- Hubara, I., Nahshan, Y., Hanani, Y., Banner, R., & Soudry, D. (2021). Accurate post training quantization with small calibration sets. In M. Meila & T. Zhang (Eds.), *Proceedings of the 38th International Conference on Machine Learning* (pp. 4466–4475, Vol. 139). PMLR. <https://proceedings.mlr.press/v139/hubara21a.html>
- Karras, T., Laine, S., & Aila, T. (2018). A style-based generator architecture for generative adversarial networks [Version Number: 3]. <https://doi.org/10.48550/arXiv.1812.04948>
- Kavalerov, I., Czaja, W., & Chellappa, R. (2021). A multi-class hinge loss for conditional GANs. *Proceedings of the 2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 1289–1298. <https://doi.org/10.1109/WACV48630.2021.00133>
- Kossale, Y., Airaj, M., & Darouichi, A. (2022). Mode collapse in generative adversarial networks: An overview. *2022 8th International Conference on Optimization and Applications (ICOA)*, 1–6. <https://doi.org/10.1109/ICOA55659.2022.9934291>
- Krizhevsky, A., Nair, V., & Hinton, G. (2009, April). *Learning multiple layers of features from tiny images* (tech. rep.). Department of Computer Science,

- University of Toronto. Toronto, ON, Canada. <http://www.cs.toronto.edu/~kriz/cifar.html>
- Li, H., Wang, Z., Yue, X., Wang, W., Tomiyama, H., & Meng, L. (2023). An architecture-level analysis on deep learning models for low-impact computations. *Artif. Intell. Rev.*, 56(3), 1971–2010. <https://doi.org/10.1007/s10462-022-10221-5>
- Lin, J., Chen, W.-M., Cai, H., Gan, C., & Han, S. (2021). MCUNetV2: Memory-efficient patch-based inference for tiny deep learning. *NIPS'21: Proceedings of the 35th International Conference on Neural Information Processing Systems*.
- Lukač, L. (2026, April). HWCD: HandWritten Character Dataset. <https://doi.org/10.5281/ZENODO.19553096>
- Majidpour, J., Ahmed, H. A., Ahmed, M. H., Jalal, S. I., & Arabi, H. (2026). Applications of GAN models in breast cancer detection: A comprehensive review. *Arch. Computat. Methods Eng.*, 33(1), 859–915. <https://doi.org/10.1007/s11831-025-10323-7>
- Metz, L., Poole, B., Pfau, D., & Sohl-Dickstein, J. (2016, November). Unrolled generative adversarial networks. <https://doi.org/10.48550/arXiv.1611.02163>
- Micikevicius, P., Narang, S., Alben, J., Diamos, G., Elsen, E., Garcia, D., Ginsburg, B., Houston, M., Kuchaiev, O., Venkatesh, G., & Wu, H. (2017). Mixed precision training. <https://doi.org/10.48550/arXiv.1710.03740>
- Mirza, M., & Osindero, S. (2014, November). Conditional generative adversarial nets. <https://doi.org/10.48550/arXiv.1411.1784>
- Nagel, M., Fournarakis, M., Amjad, R. A., Bondarenko, Y., van Baalen, M., & Blankevoort, T. (2021). A white paper on neural network quantization. <https://doi.org/10.48550/arXiv.2106.08295>
- Nguyen, T., Le, T., Vu, H., & Phung, D. (2017). Dual discriminator generative adversarial nets. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *NIPS'17: Proceedings of the 31st International Conference on Neural Information Processing Systems* (Vol. 30). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2017/file/e60e81c4cbe5171cd654662d9887aec2-Paper.pdf
- Qin, D., Bu, J.-J., Liu, Z., Shen, X., Zhou, S., Gu, J.-J., Wang, Z.-H., Wu, L., & Dai, H.-F. (2021). Efficient medical image segmentation based on knowledge distillation. *IEEE Trans. Med. Imaging*, 40(12), 3820–3831. <https://doi.org/10.1109/TMI.2021.3098703>
- Rokh, B., Azarpeyvand, A., & Khanteymoori, A. (2023). A comprehensive survey on model quantization for deep neural networks in image classification. *ACM Trans. Intell. Syst. Technol.*, 14(6), 1–50. <https://doi.org/10.1145/3623402>
- Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., Chen, X., & Chen, X. (2016). Improved techniques for training GANs. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, & R. Garnett (Eds.), *NIPS'16: Proceedings of the 30th International Conference on Neural Information Processing Systems* (Vol. 29). Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2016/file/8a3363abe792db2d8761d6403605aeb7-Paper.pdf
- Vanhala, M., & Kurvinen, E. (2026). A generative adversarial network-based methodology in systematic machine design at concept phase for solutions. *Adv. Mech. Eng.*, 18(2), 16878132261420739. <https://doi.org/10.1177/16878132261420739>
- Wang, P., & Sertel, E. (2023). Multi-frame super-resolution of remote sensing images using attention-based GAN models. *Knowl.-Based Syst.*, 266, 110387. <https://doi.org/10.1016/j.knosys.2023.110387>
- Wang, P., Wang, D., Ji, Y., Xie, X., Song, H., Liu, X., Lyu, Y., & Xie, Y. (2019). QGAN: Quantized generative adversarial networks. <https://doi.org/10.48550/arXiv.1901.08263>
- Wang, S., & Kanwar, P. (2019, August). BFloat16: The secret to high performance on cloud TPUs. Retrieved May 14, 2026, from <https://cloud.google.com/blog/products/ai-machine-learning/bfloat16-the-secret-to-high-performance-on-cloud-tpus>
- Wei, L., Ma, Z., Yang, C., & Yao, Q. (2024). Advances in the neural network quantization: A comprehensive review. *Appl. Sci.*, 14(17), 7445. <https://doi.org/10.3390/app14177445>
- Wiatrak, M., Albrecht, S. V., & Nystrom, A. (2019). Stabilizing generative adversarial networks: A survey. <https://doi.org/10.48550/arXiv.1910.00927>
- Yang, J., Shen, X., Xing, J., Tian, X., Li, H., Deng, B., Huang, J., & Hua, X.-s. (2019). Quantization networks. *Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7300–7308. <https://doi.org/10.1109/CVPR.2019.00748>
- Zhang, X.-Y., Zhu, Q.-X., Ke, W., He, Y.-L., Zhang, M.-Q., & Xu, Y. (2025). Regression loss-assisted conditional style generative adversarial network for virtual sample generation with small data in soft sensing. *Eng. Appl. Artif. Intell.*, 147, 110306. <https://doi.org/10.1016/j.engappai.2025.110306>
- Zhang, Z., Zeng, Y., Bai, L., Hu, Y., Wu, M., Wang, S., & Hancock, E. R. (2020). Spectral bounding: Strictly satisfying the 1-Lipschitz property for generative adversarial networks. *Pattern Recognit.*, 105, 107179. <https://doi.org/10.1016/j.patcog.2019.107179>