RealMixGAN: Accelerating GAN Convergence by Leaking Real Data Distribution into the Latent Space

Ankit Sen, Arghya Daw, Subhadip Malakar, Sayan Biswas
Under mentorship of **Dr. Sumanta Ray** — IEEE Summer Internship 2025

1 Introduction

Traditional GANs [1] achieve remarkable success on MNIST, CIFAR-10, and CelebA by mapping noise vectors to realistic images through adversarial feedback. However, this pure-noise paradigm suffers from unstable training, slow convergence, and mode collapse when modeling high-dimensional distributions without direct real data guidance.

RealMixGAN introduces a paradigm shift by "leaking" controlled real data portions into the generator's input. Instead of relying exclusively on random noise, RealMixGAN augments latent samples with noisy projections of real images from a rolling buffer. Both networks use dropout regularization [4] and tuned Adam optimizer (betas = 0.5, 0.999) for stability. By fusing genuine data statistics with random noise under high mixing probability, RealMixGAN expedites convergence, enhances mode coverage, and yields smoother adversarial losses.

2 Proposed Method

RealMixGAN injects real data samples into the generator's latent input under probabilistic control while preserving standard adversarial loss.

Real-Image Buffer: Rolling buffer capacity of 1,000 flattened real images (784-dim) with 10% refresh rate per iteration for diversity maintenance.

Latent Mixing Module: Mix probability $p_{\text{mix}} = 0.90$ draws minibatches from buffer. Gaussian noise ($\sigma = 0.2$) is added to buffer samples, clamped to $[-1, 1]$. Mixing ratio $\alpha = 0.20$ multiplies noisy real vectors, concatenated with 100-dim random noise z, producing 884-dim generator input. Otherwise (10%), concatenate z with fresh Gaussian noise ($\sigma = 0.1$).

Table 1: Architecture Comparison

Component	**Standard GAN**	**RealMixGAN**
Generator Input	100	884
Generator	100→256→512→1024→784	884→256→512→1024→784
Discriminator	784→1024→512→256→1	784→1024→512→256→1
Dropout	None	20% (Gen), 30% (Disc)
Adam Betas	(0.9, 0.999)	(0.5, 0.999)
Learning Rate	2×10^{-4}	2×10^{-4}

Key Improvements: Dropout regularization prevents overfitting; real-data influence ensures better mode coverage; noise-injection buffer preserves stochasticity; beta-parameter tuning enables smoother updates; adaptive buffer management maintains diversity.

Algorithm 1: RealMixGAN Training

Input: Dataset $D = \{x_i\}_1^n \subset \mathbb{R}^d$ (d=784), batch size B, latent dim z_{dim}, mix probability $p_{mix} \in [0, 1]$, mix ratio $\alpha \in [0, 1]$, noise scale σ, buffer capacity C_{buf}, learning rate η, Adam (β_1, β_2), epochs N_{epochs}

Initialize: Generator θ_G, Discriminator θ_D, buffer $\mathbb{B} \leftarrow \emptyset$

for epoch = 1 to N_{epochs} **do**
  for each minibatch $\{x_i\}_{i=1}^B$ from D **do**
    1. **Buffer update:** If $|\mathbb{B}| < C_{buf}$, add $\lfloor B/4 \rfloor$ samples to $\mathbb{B}$; else replace random subset
    2. **Sample labels:** $y_{real} = 1^B$, $y_{fake} = 0^B$
    3. **Discriminator step:**
      a. $L_{D_{real}} = \mathbb{E}_x[\text{BCE}(D(x), y_{real})]$
      b. Sample $z \sim \mathcal{N}(0, I) \in \mathbb{R}^{B \times z_{dim}}$
      c. With prob. p_{mix} and $|\mathbb{B}| \geq B$: Sample $\{x_{buf}\}$ from $\mathbb{B}$,
        $x_{noisy} = \text{clamp}(x_{buf} + \sigma \cdot \epsilon, -1, 1)$, input $= [z, \alpha \cdot x_{noisy}]$
        Else: input $= [z, r]$ where $r \sim \mathcal{N}(0, 0.1^2 I)$
      d. $x_{fake} = G(\text{input}; \theta_G)$, $L_{D_{fake}} = \mathbb{E}[\text{BCE}(D(x_{fake}), y_{fake})]$
      e. $L_D = L_{D_{real}} + L_{D_{fake}}$, Update $\theta_D \leftarrow \theta_D - \eta \nabla_{\theta_D} L_D$
    4. **Generator step:** $L_G = \mathbb{E}[\text{BCE}(D(x_{fake}), y_{real})]$, Update $\theta_G \leftarrow \theta_G - \eta \nabla_{\theta_G} L_G$
  end for
end for

3 Experimental Results

Both Standard GAN and RealMixGAN trained on MNIST for 70 epochs (batch size = 128), evaluated via Wasserstein distance [3] and loss stability.

Table 2: Performance Metrics

Metric	Standard GAN	RealMixGAN
Final Wasserstein Distance	0.0290	**0.0137 (52.8% ↓)**
Generator Loss (Epoch 70)	1.3177	**1.0261 (22.1% ↓)**
Discriminator Loss (Epoch 70)	1.0431	1.2033

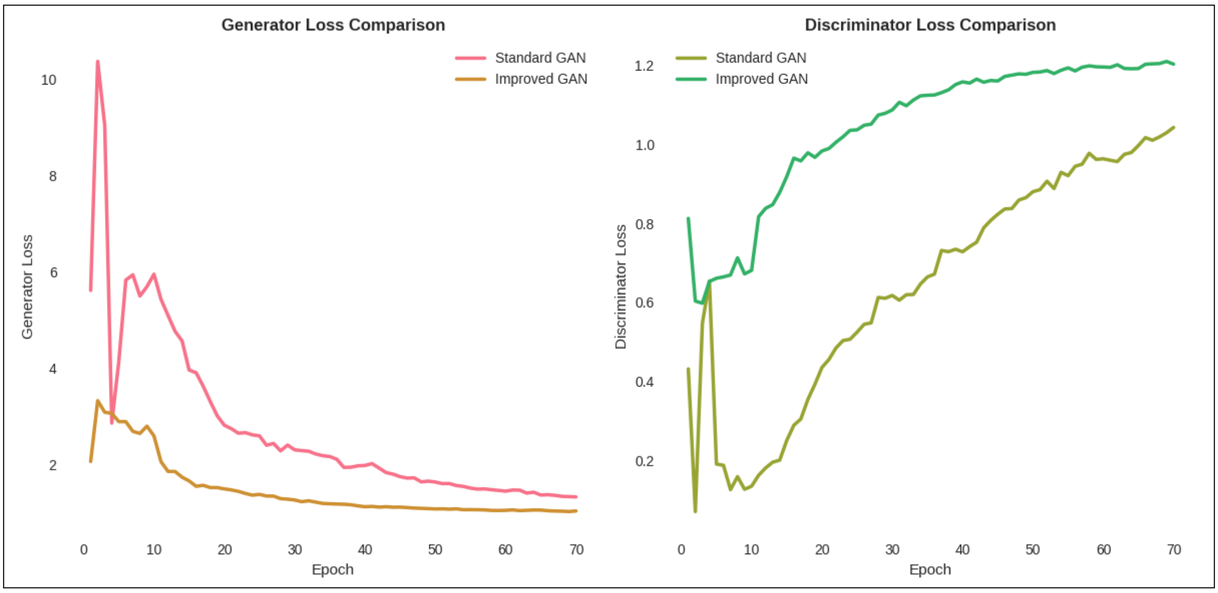


Figure 1: Generator and Discriminator Loss Comparison

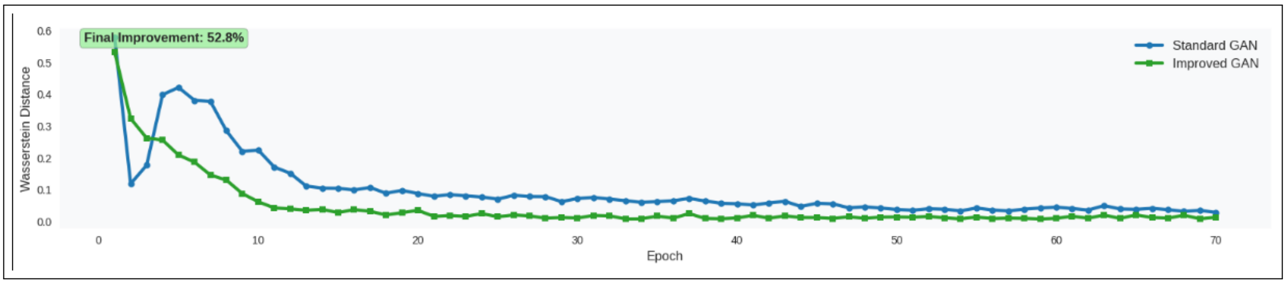
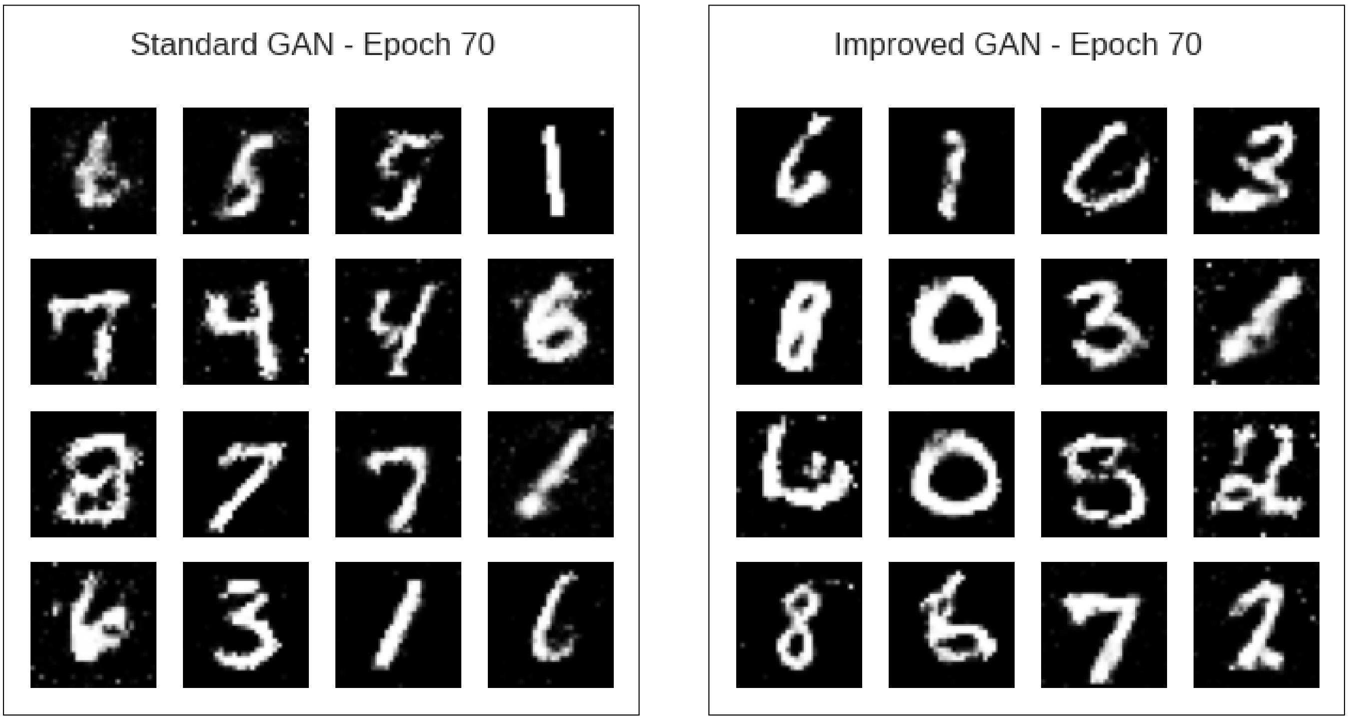


Figure 2: Wasserstein Distance Evolution showing improved convergence

Training Stability Analysis: RealMixGAN demonstrates markedly smoother loss trajectories with fewer oscillations, indicating stable adversarial dynamics. The combination of dropout, real-data mixing, and tuned Adam betas prevents overfitting and mitigates mode collapse. Extended training over 70 epochs reveals consistent convergence patterns with RealMixGAN maintaining superior performance throughout the training process.



(a) Standard GAN Samples

(b) RealMixGAN Samples

Figure 3: Generated MNIST samples comparison

4 Conclusion

RealMixGAN extends vanilla GAN by "leaking" controlled real data fractions into generator latent input. Through efficient Real-Image Buffer and probabilistic Mixing Module—coupled with dropout regularization and tuned hyperparameters—RealMixGAN achieves substantially faster convergence and richer sample diversity.

MNIST results show 52.8% Wasserstein distance reduction, 22.1% generator loss decrease, and robust discriminator performance. Smoother loss curves confirm our data-leakage mechanism prevents mode collapse while accelerating training.

RealMixGAN's hybrid noise + data prior offers generalizable strategy for improving generative modeling. Future work explores convolutional extensions, adaptive mixing schedules, and complex dataset applications for enhanced scalability.

References

[1] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, "Generative Adversarial Networks," in *Advances in Neural Information Processing Systems*, 2014.

[2] Alec Radford, Luke Metz, Soumith Chintala, "Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks," in *International Conference on Learning Representations*, 2016.

[3] Martin Arjovsky, Soumith Chintala, Léon Bottou, "Wasserstein Generative Adversarial Networks," in *International Conference on Machine Learning*, 2017.

[4] Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, "Improved Techniques for Training GANs," in *Advances in Neural Information Processing Systems*, 2016.