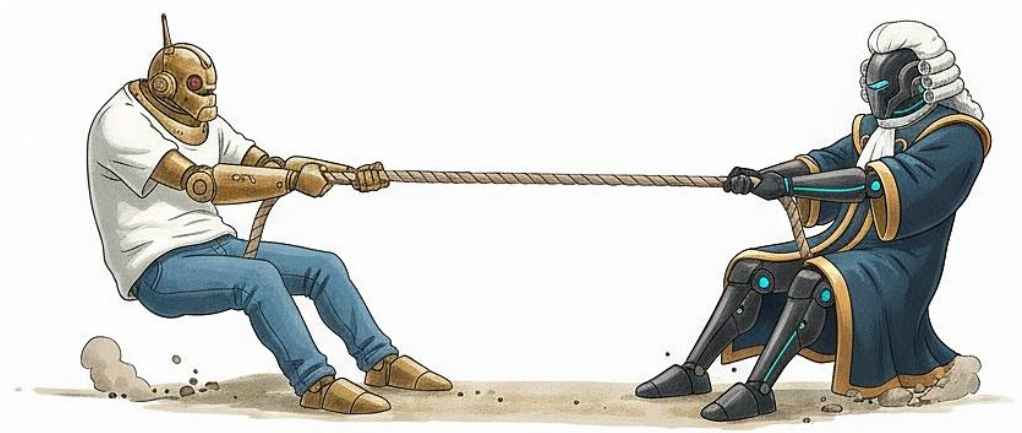
TAAP

INSTITUTO UNIVERSITÁRIO DE LISBOA



Advanced Topics in Deep Learning

MINI-PROJECT 01

IMPLEMENTING A CONDITIONAL GENERATIVE ADVERSARIAL NETWORK

António Cruz (140129), **Cátia Brás** (120093), **Ricardo Kayseller** (95813)

Table of Contents

1. Introduction

Generative Adversarial Networks (GANs) have become a fundamental class of generative models for learning complex data distributions and synthesizing realistic samples. In the classical (unconditional) GAN framework, the generator receives only random noise as input and learns to produce samples that resemble the training data, while the discriminator learns to distinguish real data from generated samples. Although this setting allows the model to generate realistic images, it provides no control over the semantic content of the generated samples. In the context of datasets with labeled structure, such as MNIST, an unconditional GAN may generate any digit between 0 and 9, but the user cannot specify which digit should be produced.

Conditional Generative Adversarial Networks (cGANs) extend the original GAN formulation by incorporating additional side information, such as class labels, into both the generator and the discriminator. By conditioning the generation process on a label y, the generator learns a mapping $G(z,y)$ that aims to produce samples consistent with the desired class, while the discriminator is trained to assess not only whether an image is real or fake, but also whether it matches the provided condition.

This conditioning mechanism enables controlled generation and significantly increases the practical usefulness of GANs in applications where semantic attributes matter, such as digit synthesis, object class control, and, more generally, text-to-image generation.

In this project, we investigate the implementation and behavior of conditional GANs on the MNIST dataset of handwritten digits. Starting from an unconditional DCGAN baseline, we progressively introduce conditioning mechanisms that allow explicit control over the generated digit class. Beyond basic conditioning via concatenation of labels, we explore stabilization strategies inspired by the literature, such as Spectral Normalization applied to the discriminator, in order to mitigate training instabilities and mode collapse. The experimental analysis focuses on both qualitative and quantitative aspects, including visual inspection of generated samples, class-conditional control checks, intra-class diversity, and discriminator behavior during training.

The main objective of this work is to demonstrate that conditional adversarial training enables controllable image generation, to analyze the limitations of simple conditioning strategies, and to assess how architectural and training refinements improve stability, diversity, and label consistency. This study provides practical insights into the challenges of training cGANs and highlights the importance of appropriate conditioning and regularization techniques for achieving reliable class-conditional generation.

2. Environment Setup & Configuration

2.1 Global Variables and Training Hyperparameters

```
SEED = 42

# Mode: True for cGAN (conditional), False for DCGAN (unconditional)
CONDITIONAL = True

LIVE_MONITOR = False
LIVE_MONITOR_PORT_NUMBER = 8992
EMIT_INTERVAL = 1

DATASET_PATH = "../../dataset/"

BATCH_SIZE = 128
LATENT_DIM = 100
NUM_CLASSES = 10

# Path to calibrated classifier checkpoint
CLASSIFIER_CHECKPOINT = "classifier/model/mnist_cnn_best.ckpt"

# E.g.: NUM_STEPS = 15005 = ~32 epochs (60000 images / 128 batch_size ≈ 469 steps/epoch)
NUM_STEPS = 30010
SAVE_INTERVAL = 1000

# TTUR: Two Time-Scale Update Rule - D learns faster than G, so use different learning rates
LR_D = 4e-4  # Discriminator learning rate
LR_G = 1e-4  # Generator learning rate (4x slower)

# Adam betas optimized for GAN training
# Lower β1 (0.5 vs default 0.9) reduces momentum, stabilizes adversarial updates
ADAM_BETAS = (0.5, 0.999)

# Label smoothing for BCE/LSGAN (use 0.9 instead of 1.0 for real labels)
LABEL_SMOOTHING_REAL = 0.9

# WGAN-GP: number of critic steps per generator step
N_CRITIC = 5

# WGAN-GP: gradient penalty coefficient
LAMBDA_GP = 10.0

# Model output path includes mode subfolder (cgan or dcgan)
MODEL_OUTPUT_PATH = f"model/{'cgan' if CONDITIONAL else 'dcgan'}/"
D_MODEL_NAME = "D_DRAFT_01"
G_MODEL_NAME = "G_DRAFT_01"

NUM_EVAL_SAMPLES = 10000

# Strategies to benchmark
BENCHMARK_STRATEGIES = ["bce", "lsgan", "hinge", "wgan-gp"]
```

2.2 Setup and Training Strategy

This project adopts a systematic experimental design to analyze the behavior and stability of conditional Generative Adversarial Networks (cGANs) on the MNIST dataset. Beyond implementing a single adversarial formulation, multiple loss strategies are benchmarked, including the standard binary cross-entropy (BCE) loss, Least Squares GAN (LSGAN), hinge loss, and Wasserstein GAN with Gradient Penalty (WGAN-GP). This design choice follows the theoretical and empirical insights discussed in the lecture materials on GAN losses and training stability, which highlight that different adversarial objectives lead to substantially different optimization dynamics, convergence behavior, and robustness to mode collapse. In particular, WGAN-GP is included due to its smoother loss landscape and explicit enforcement of the Lipschitz constraint, which is known to improve training stability compared to classical GAN formulations.

Training is performed using the Two Time-Scale Update Rule (TTUR), where the discriminator is updated with a learning rate four times higher than the generator (LR_D = 4×10^{-4}, LR_G = 1×10^{-4}). This choice is directly motivated by best practices discussed in the GAN training "recipe" guidelines, which emphasize that an overly weak discriminator fails to provide informative gradients to the generator, while an overly strong discriminator can lead to vanishing gradients. The Adam optimizer is used with $\beta_1 = 0.5$ and $\beta_2 = 0.999$, following the DCGAN and subsequent GAN literature, as these hyperparameters are empirically known to stabilize adversarial training by reducing excessive momentum in the discriminator updates.

The batch size is set to 128, representing a compromise between training stability and computational efficiency. Larger batch sizes generally provide more stable gradient estimates for both generator and discriminator, while still fitting comfortably within GPU memory constraints. The dimensionality of the latent space is fixed to 100, following common practice in DCGAN-style architectures, which provides sufficient capacity for modeling the variability of handwritten digits without introducing unnecessary complexity.

Training is defined in terms of a fixed number of optimization steps rather than epochs. Specifically, the model was trained for 30,010 steps. This value corresponds approximately to 64 epochs over the MNIST training set (MNIST contains 60,000 samples, and with a batch size of 128, one epoch corresponds to roughly 469 iterations).

The step-based formulation is consistent with how GAN experiments are typically reported in the literature, as it provides finer control over the balance between generator and discriminator updates, enables predictable checkpointing and sampling intervals, and facilitates fair comparisons across different training configurations. The additional offset of five steps ensures that the final checkpoint aligns exactly with the predefined sampling interval of 1,000 steps, allowing the evolution of generated samples to be consistently monitored at fixed milestones throughout training.

2.3 Real-Time Monitoring Dashboard

A real-time monitoring dashboard was developed and integrated into the training pipeline to support continuous inspection of adversarial dynamics during benchmarking. When

enabled, a lightweight local server streams training diagnostics such as generator sample grids, discriminator and generator loss trajectories, and per-strategy progress markers.

This design aligns with best-practice recommendations for GAN training, where losses alone are often insufficient to assess convergence or detect failure modes. The live monitor improves experimental transparency, facilitates early detection of mode collapse or imbalance between networks, and ensures consistent reporting across multiple loss strategies within the same benchmarking framework.

3. GAN Loss Strategies

3.1 Benchmarking Framework

In order to systematically study the impact of different adversarial objectives on training stability and sample quality, the training pipeline was designed around a modular loss-strategy framework. Instead of hard-coding a single GAN loss, a common abstract interface was defined to encapsulate the discriminator and generator loss components, enabling multiple loss formulations to be benchmarked under identical architectural and optimization conditions.

Four widely adopted loss strategies were implemented and evaluated:

1. **Binary Cross-Entropy (BCE) Loss**

   This strategy corresponds to the original GAN formulation, where the discriminator is trained as a probabilistic classifier distinguishing real from generated samples. The generator is optimized to maximize the probability that fake samples are classified as real. To improve training stability, label smoothing is applied to real samples, reducing overconfidence in the discriminator and mitigating sharp gradients that may destabilize the adversarial dynamics. This formulation directly reflects the classical minimax game described in the original GAN literature.

2. **Least Squares GAN (LSGAN)**

   The LSGAN objective replaces the binary cross-entropy loss with a least-squares regression loss. Instead of learning to output hard binary decisions, the discriminator is encouraged to regress towards continuous target values for real and fake samples. This formulation has been shown to reduce vanishing gradients and produce smoother optimization landscapes, often resulting in more stable convergence during training. As in the BCE strategy, label smoothing is applied to the real targets to further regularize the discriminator.

3. **Hinge Loss**

   The hinge loss formulation adopts a margin-based objective, where the discriminator enforces a separation between real and fake samples using a hinge function. Rather than predicting explicit probabilities, the discriminator outputs real-valued scores, and the generator is trained to maximize these scores for generated samples. This loss is commonly used in modern high-performance GAN architecture and aligns with the design patterns discussed in contemporary GAN literature, particularly in combination with architectural constraints such as spectral normalization.

4. **Wasserstein GAN with Gradient Penalty (WGAN-GP)**

   The WGAN-GP strategy reformulates the adversarial game in terms of the Wasserstein distance between the real and generated data distributions. Instead of a classifier, the discriminator acts as a critic that assigns scalar scores to samples. To enforce the required Lipschitz continuity constraint, a gradient penalty term is added, penalizing deviations of the gradient norm from unity along linear

interpolations between real and fake samples. Additionally, the critic is updated multiple times per generator update, following the two time-scale update principle, to ensure a sufficiently strong approximation of the Wasserstein distance during training.

Further experiments indicated that the degradation observed in the initial WGAN-GP configuration was strongly linked to the simultaneous application of multiple regularization mechanisms.

In particular, the combination of Spectral Normalization and Gradient Penalty imposed redundant Lipschitz constraints on the Discriminator, while Dropout further reduced its effective capacity. In WGAN-GP, the discriminator is designed without spectral normalization or dropout, given that the gradient penalty already ensures the Lipschitz constraint, while stochastic components may interfere with the stability and reliability of the penalty term.

By implementing these loss strategies within a unified interface, the project enables a fair and controlled comparison of fundamentally different adversarial objectives. All strategies share the same generator and discriminator architectures, optimization settings, and training schedule, ensuring that observed differences in convergence behavior, stability, and sample quality can be attributed primarily to the choice of loss function rather than to confounding implementation details.

This benchmarking setup reflects the theoretical and practical considerations emphasized in the course materials, where the choice of adversarial loss is a central factor in the stability and effectiveness of GAN training.

3.2 Software Architecture

To enable systematic benchmarking of multiple adversarial objectives and network configurations within a single codebase, the training framework was designed around two core software patterns:

- **Strategy Pattern**: All loss functions inherit from an abstract GANLossStrategy base class that defines a common interface (d_loss_real, d_loss_fake, g_loss, gradient_penalty). This allows BCE, LSGAN, Hinge, and WGAN-GP to be used interchangeably without modifying the training loop.
- **Factory Pattern**: The functions get_loss_strategy(), get_discriminator(), and get_adam_betas() instantiate the appropriate components based on the selected strategy. Notably, WGAN-GP uses a specialized DiscriminatorWGAN architecture (with LayerNorm, no Spectral Normalization or Dropout) since the gradient penalty already enforces the Lipschitz constraint.
- **Conditional Flag**: A global CONDITIONAL parameter switches between cGAN mode (label-conditioned generation) and standard DCGAN mode (unconditional). Both Generator and Discriminator classes accept an optional labels argument, enabling the same architecture to support both configurations.

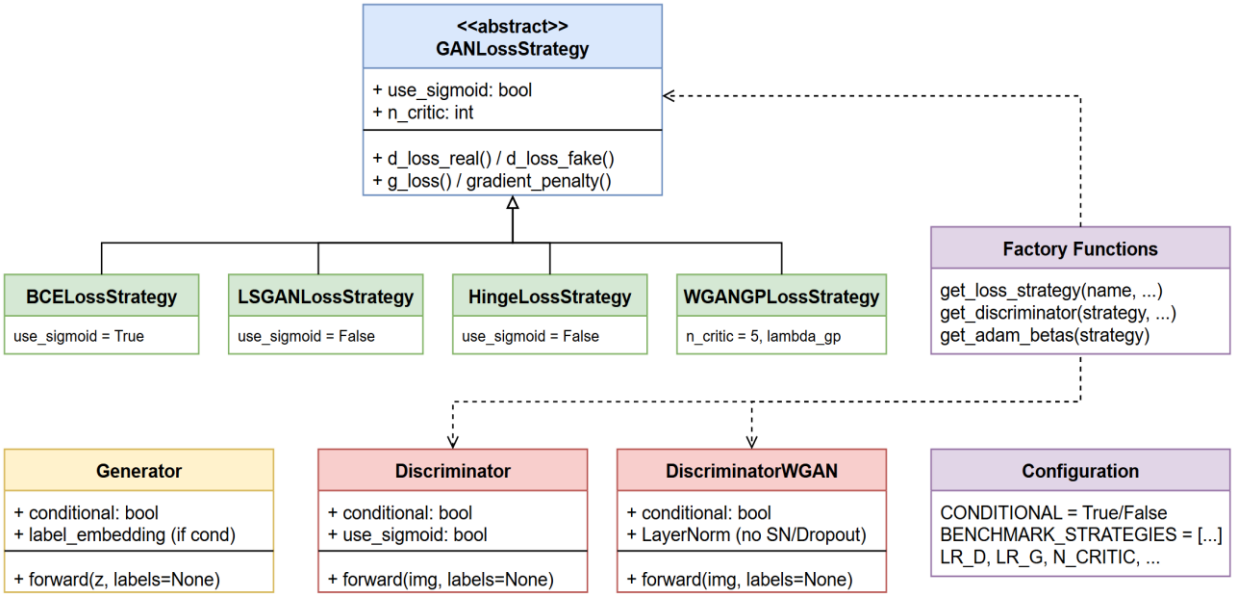


Figure 3.1 - Class architecture of the GAN training framework. The Strategy pattern enables interchangeable loss functions, while Factory functions create strategy-appropriate discriminators. The CONDITIONAL flag toggles between cGAN and DCGAN modes.

4. Data Loading

4.1 Data Loading and Preprocessing

The MNIST dataset was loaded using the PyTorch torchvision.datasets interface, ensuring seamless integration with the training pipeline and compatibility with GPU-accelerated mini-batch processing. All experiments were conducted using the official training split of MNIST, which contains 60,000 grayscale images of handwritten digits (28×28 resolution), each annotated with a class label in the range {0,…,9}. The test split was not used during adversarial training, as GAN optimization is inherently unsupervised with respect to distribution matching.

Prior to training, a deterministic preprocessing pipeline was applied to all images. First, each image was converted from a PIL object to a floating-point tensor using transforms.ToTensor(). This operation rescales pixel intensities from the integer range [0,255] to the normalized interval [0,1]. Subsequently, the tensor was normalized using a mean of 0.5 and a standard deviation of 0.5, mapping values from [0,1] to the symmetric range [-1,1] according to the transformation:

$$x_{norm} = \frac{x - 0{,}5}{0{,}5} = 2x - 1$$

This normalization step is essential for stable adversarial training because the Generator produces outputs through a final Tanh activation function, whose natural output domain is [-1,1]. Ensuring that real MNIST samples and generated samples lie on the same numerical scale prevents systematic bias in the Discriminator and avoids unnecessary gradient distortions during early training phases.

Data loading was performed using the PyTorch DataLoader abstraction, which provides automatic batching, optional shuffling, and efficient memory management. A batch size of 128 was used, balancing gradient stability and computational efficiency as discussed in Section 2. The training data was shuffled at the beginning of each epoch to reduce correlation between successive mini-batches and to promote more stable adversarial dynamics.

Importantly, the drop_last=True parameter was enabled. This configuration discards the final incomplete mini-batch when the dataset size is not perfectly divisible by the batch size. Although MNIST (60,000 samples) is divisible by 128 with a small remainder, enforcing a constant batch size simplifies loss computation, logging consistency, and benchmarking across adversarial objectives. In particular, certain loss formulations (e.g., WGAN-GP with gradient penalty) assume uniform tensor shapes for efficient computation of interpolated samples and gradient norms. Maintaining a fixed batch size therefore eliminates edge-case behavior and ensures that all strategies are evaluated under strictly identical conditions.

5. Generator & Discriminator

5.1 Conditional Generator

The conditional generator was implemented using a PixelShuffle-based upsampling architecture. Class conditioning was introduced through a learned embedding of the digit label into the latent space, combined multiplicatively with the noise vector to modulate the generation process.

The model first projects the conditioned latent code into a 7×7×128 feature map, then performs two stages of sub-pixel upsampling (7→14→28) using convolution + PixelShuffle blocks with Batch Normalization and ReLU activations. To reduce checkerboard and dot artifacts that can emerge from upsampling, ICNR initialization was applied to the convolutional layers preceding PixelShuffle.

Finally, the generator outputs a 28×28 grayscale image through a Tanh activation, which is consistent with preprocessing that normalizes MNIST images to the range [-1, 1].

5.2 Conditional Discriminator (Standard & WGAN-GP Variants)

The conditional discriminator was implemented using a label-map concatenation strategy, where each class label is embedded into a 28×28 spatial map and concatenated with the input image as an additional channel. This formulation allows the discriminator to jointly assess image realism and class consistency by conditioning its decision on both visual content and the target label.

For the standard adversarial objectives (BCE, LSGAN, and hinge), the discriminator architecture consists of two convolutional blocks that progressively downsample the input resolution (28×28 → 14×14 → 7×7), followed by fully connected layers that output a single authenticity score. Spectral Normalization (SN) is applied to both convolutional and linear layers to constrain the Lipschitz constant of the discriminator, which stabilizes adversarial training by reducing gradient explosions, oscillatory dynamics, and susceptibility to mode collapse. Dropout is included as a regularization mechanism to improve generalization and mitigate overfitting in the discriminator.

Depending on the selected loss function, the final output is either passed through a sigmoid (for BCE) or left as raw logits (for LSGAN and hinge), enabling each objective to apply its appropriate formulation.

For WGAN-GP, a dedicated critic architecture is employed, reflecting the methodological requirements of Wasserstein-based training. In this variant, Spectral Normalization and Dropout are explicitly removed and replaced with Layer Normalization. This design choice is motivated by three considerations:

1. **Lipschitz regularization**: Spectral Normalization and gradient penalty both enforce Lipschitz constraints; combining them results in redundant regularization and can over-constrain the critic, degrading the quality of the learned Wasserstein distance.

2. **Gradient penalty stability**: Dropout introduces stochastic perturbations in the critic's forward pass, which interferes with the computation of reliable input gradients required by the gradient penalty term.
3. **Critic capacity**: The channel width is increased (from 64 to 128 filters in deeper layers) to provide sufficient representational capacity for accurately approximating the Wasserstein distance between real and generated distributions.

The WGAN-GP critic outputs raw scalar scores without any sigmoid activation, as required by the Wasserstein formulation. The appropriate discriminator variant (standard SN-based discriminator vs. WGAN-GP critic) is selected automatically based on the chosen adversarial objective, ensuring that each loss strategy is paired with an architecture consistent with its theoretical assumptions and practical stability requirements.

5.3 Weight Initialization

Model parameters were initialized following the DCGAN guidelines, which recommend sampling convolutional and fully connected weights from a Normal(0, 0.02) distribution and initializing batch normalization layers with weights drawn from Normal(1, 0.02) and zero bias. This initialization scheme has been shown to promote stable adversarial dynamics during the early stages of training and to reduce the likelihood of vanishing gradients or premature mode collapse.

Label embedding layers were initialized using the same Normal(0, 0.02) distribution to ensure comparable scale between latent and conditional representations. For layers wrapped with Spectral Normalization, initialization was applied to the underlying un-normalized weight parameters (weight_orig) rather than the normalized weights, ensuring consistent parameter scaling at initialization while preserving the Lipschitz constraint enforced during training.

This design choice aligns with best practices in stabilizing GAN training, as discussed in the DCGAN and SNGAN literature, where careful weight initialization and spectral normalization jointly contribute to improved convergence behavior and reduced training oscillations.

6. Training & Evaluation

6.1 Class Consistency

Models were instantiated on the selected compute device, with the generator using its internal initialization (including ICNR initialization for PixelShuffle upsampling) and the discriminator initialized using DCGAN-style weight initialization to improve early training stability. A modular loss-strategy interface was used to select the adversarial objective (BCE, LSGAN, hinge, or WGAN-GP). For WGAN-GP, the discriminator instance was passed into the strategy object to enable computation of the gradient penalty term, enforcing the Lipschitz constraint required by the Wasserstein formulation.

Training used the Two Time-Scale Update Rule (TTUR), configuring separate Adam optimizers for discriminator and generator with a higher discriminator learning rate (LR_D > LR_G) and GAN-optimized momentum parameters (β_1=0.5, β_2=0.999). This setup supports stable adversarial dynamics and enables fair benchmarking across different loss formulations under consistent optimization conditions.

To complement distribution-based metrics (FID/KID) and qualitative inspection, we introduced a class-consistency evaluation to explicitly measure whether the conditional generator obeys the requested label. This metric quantifies label alignment by computing the percentage of generated images whose predicted class (from an external classifier) matches the conditioning label y. This is particularly relevant for conditional GANs on MNIST because a model can produce visually plausible digits while still ignoring conditioning information (a known failure mode in cGANs).

Metric definition

- For each class $y \in \{0, \ldots, 9\}$, we generate N samples using $G(z,y)$, then evaluate them with a pretrained MNIST classifier $C(\cdot)$.

We also report per-class consistency, which is useful to detect asymmetric failures (e.g., digits that collapse into visually similar classes such as 3/5/8 or 4/9):

- Directly targets the conditioning objective: Unlike FID/KID, this metric evaluates whether the generator respects the provided label.
- Detects “conditional collapse”: The generator may learn a narrow subset of digits and still appear stable, but class-consistency immediately exposes label leakage or label ignoring.
- MNIST-appropriate: A digit classifier is trained on the same domain (handwritten digits), making it a strong and interpretable control metric.

We used the same MNIST classifier employed previously (in our previous MNIST project) to ensure consistency across experiments. The classifier architecture (a custom CNN) was re-instantiated without Lightning dependencies, and the weights were loaded from a Lightning-style checkpoint (state_dict).

Before using the classifier for evaluation, we verified that it achieved high accuracy on real MNIST to ensure it is a reliable oracle for class-consistency. If the checkpoint is missing, the evaluation is safely skipped (to avoid breaking the benchmark pipeline).

This evaluation was applied to each trained generator configuration (different loss strategies or regularization setups), enabling a clear comparison of how each strategy affects label compliance, not only visual fidelity.

6.2 Benchmarking

Under a controlled setup (same dataset, generator/discriminator family, TTUR, and step budget), the benchmark compared four adversarial objectives using three evaluation axes: distributional fidelity (FID, KID), conditional correctness (Class-Consistency), and computational cost (training time).

Overall ranking

- **WGAN-GP** achieved the best quantitative performance: FID = 4.79, KID = 0.0028 ± 0.0015, with the highest Class-Consistency = 98.90%, but it was also the slowest (1857s).
- **BCE** and **LSGAN** formed a strong middle tier with similar class-consistency and runtime, achieving competitive fidelity:
  - **BCE**: FID = 5.06, KID = 0.0024 ± 0.0020, Class-Consistency = 97.90%, 495s
  - **LSGAN**: FID = 5.22, KID = 0.0028 ± 0.0020, Class-Consistency = 98.46%, 513s
- **Hinge** underperformed in fidelity while keeping class-consistency comparable:
  - **Hinge**: FID = 6.84, KID = 0.0033 ± 0.0021, Class-Consistency = 97.80%, 502s

What do the numbers suggest:

1. **Fidelity (FID/KID): WGAN-GP leads, but BCE/LSGAN are close**

   WGAN-GP's FID 4.79 and KID 0.0028 indicate the closest alignment between generated and real sample distributions among all strategies tested. This is consistent with the Wasserstein objective, which provides smoother, more informative gradients than probability-based discriminators, and with the gradient penalty, which encourages a well-behaved critic function.

   Notably, BCE and LSGAN achieve strong results (FID 5.06 and 5.22 respectively), narrowing the gap to WGAN-GP compared to earlier configurations. BCE achieves the lowest KID (0.0024), indicating excellent second-order distributional alignment. Hinge lags behind at FID 6.84, suggesting that in this architecture/regime, margin-based optimization did not translate into improved distributional match.

2. **Conditional correctness: all strategies achieve high consistency**

   Class-consistency is high across all objectives (97.80%–98.90%), meaning that conditioning works reliably: the generator typically produces a digit that matches the requested label.

WGAN-GP achieves the highest consistency (98.90%), followed closely by LSGAN (98.46%), BCE (97.90%), and Hinge (97.80%). The uniformly high scores indicate that the conditioning mechanism is effective regardless of the adversarial objective, with WGAN-GP providing marginally tighter coupling between label-conditioning and generated structure.

3. **Efficiency: WGAN-GP is ~3.7× slower**

   WGAN-GP took 1857s, versus ~495–513s for the other losses. This is fully expected because:

   - It uses multiple critic updates per generator update (*n_critic* = 5).
   - Computes gradient penalty, which requires additional forward and backward passes through the critic on interpolated samples.

So WGAN-GP provides the best quality here, but at a significant computational cost.

Interpreting the loss curves

Loss magnitudes are not directly comparable across objectives because the losses represent different quantities:

- **BCE / LSGAN** losses are tied to probabilistic classification / regression-to-targets (their absolute values remain in relatively bounded ranges). BCE stabilizes around 0.8, LSGAN around 0.33.
- **Hinge** uses margin penalties; its generator loss oscillates around zero throughout training, reflecting the margin-based formulation.
- **WGAN-GP** losses represent critic score differences plus penalty; the generator loss becoming strongly negative (reaching approximately -34) is not inherently "bad"—it indicates the critic assigns high scores to generated samples and must be interpreted alongside FID/KID and visual samples.

Given that WGAN-GP achieves the best FID/KID and class-consistency, its loss trajectory is consistent with effective training rather than instability.

Key Takeaways

1. **Best quality**: WGAN-GP achieved the strongest fidelity and conditional correctness (FID 4.79, KID 0.0028, 98.90% consistency).
2. **Best speed/quality trade-off**: BCE offers excellent results (FID 5.06, 97.90% consistency) with the fastest training time (495s). LSGAN performs comparably (FID 5.22, 98.46% consistency, 513s).
3. **Hinge not ideal here**: Hinge produced worse fidelity (FID 6.84) without improving conditional correctness, suggesting it may require different regularization/capacity or a different discriminator configuration to be competitive in this setup.
4. **Compute matters**: WGAN-GP's gains come at ~3.7× training time, which is important when selecting a practical default strategy for resource-constrained scenarios.

6.3 Benchmarking Summary

6.3.1 Comparative Analysis of Adversarial Loss Strategies

This section provides a critical comparison of four adversarial objectives: **BCE**, **LSGAN**, **Hinge**, and **WGAN-GP**, grounded both in the empirical benchmark results and in the theoretical motivations discussed in the course materials (GAN objectives, stability considerations, and troubleshooting guidelines).

Strategy	FID	KID	Time (s)	Class-Consistency
wgan-gp	4.79	0.0028 ± 0.0015	1856.6	98.90%
bce	5.06	0.0024 ± 0.0020	495.4	97.90%
lsgan	5.22	0.0028 ± 0.0020	512.7	98.46%
hinge	6.84	0.0033 ± 0.0021	501.7	97.80%

Table 6.1 - Performance evaluation of different loss functions showing Fréchet Inception Distance (FID), Kernel Inception Distance (KID), training time, and class-consistency metrics.

Lower FID and KID values correspond to a closer match between the generated and real data distributions. Training time reflects the practical computational cost associated with each objective.

6.3.2 Theoretical Interpretation and Practical Implications

WGAN-GP achieved the strongest performance in terms of FID and class-consistency, which is consistent with the theoretical foundations of the Wasserstein distance. As discussed in the lecture materials on GAN stability and loss functions, Wasserstein objectives provide smoother gradients even when the discriminator (critic) is near optimal, alleviating vanishing gradient issues that commonly arise in the original BCE-based GAN formulation.

The inclusion of a **gradient penalty** further enforces the Lipschitz constraint required for the Wasserstein formulation, leading to more stable optimization dynamics and improved sample fidelity.

However, these benefits come at a substantial computational cost: WGAN-GP required approximately **3.7x longer training time** than the other strategies due to (i) multiple critic updates per generator step (*n_critic* = 5) and (ii) the additional backward passes needed to compute the gradient penalty. This confirms the trade-off highlighted in the course notes: while WGAN-GP is often more stable and produces higher-quality samples, it is significantly more expensive to train and may be impractical in resource-constrained settings.

BCE and **LSGAN** formed a strong middle tier, achieving very similar FID scores (5.06 and 5.22 respectively) and class-consistency (~98%), while requiring only about one-quarter of WGAN-GP's training time. Notably, **BCE achieved the lowest KID (0.0024)**, suggesting excellent second-order distributional alignment. The competitive performance of both objectives aligns with the motivation presented in the lectures: when combined with

modern stabilization techniques (TTUR, Spectral Normalization, conditional inputs), classical GAN losses remain highly effective for low-resolution conditional generation tasks.

Hinge loss, commonly adopted in large-scale architectures such as SAGAN and BigGAN, underperformed in this benchmark. Despite achieving class-consistency comparable to BCE (97.80%), Hinge exhibited noticeably worse FID (6.84) and KID (0.0033).

This outcome suggests that margin-based objectives may require either higher model capacity, stronger regularization, or larger and more complex datasets to fully realize their advantages. In the present low-resolution MNIST setting, the hinge formulation did not translate into improved distributional alignment, echoing the course discussion that no single GAN loss is universally optimal across domains and scales.

6.3.3 Stability Considerations and Architecture–Loss Interactions

The observed performance differences must also be interpreted in light of the discriminator design choices discussed in the lectures on GAN stabilization. In particular, the interaction between **Spectral Normalization, gradient penalty, and regularization mechanisms (e.g., Dropout or Layer Normalization)** can significantly influence training dynamics.

A key architectural decision in this implementation was the use of a **dedicated DiscriminatorWGAN class** for the WGAN-GP objective. This variant removes Spectral Normalization and Dropout (which conflict with gradient penalty computation) and replaces them with LayerNorm, while also increasing model capacity (64→128 filters). The strong performance of WGAN-GP in this configuration validates the theoretical claim that explicitly enforcing Lipschitz continuity via gradient penalty requires careful architectural adaptation to avoid over-regularization.

Conversely, the competitive performance of BCE and LSGAN highlights that simpler objectives, when combined with appropriate architectural and optimization heuristics (e.g., TTUR, Spectral Normalization, careful initialization), can achieve stable training and high-quality results at a fraction of the computational cost.

6.3.4 Overall Assessment

The benchmark confirms several key points emphasized in the course materials:

- **There is a clear quality–efficiency trade-off**: WGAN-GP delivers the highest sample fidelity (FID 4.79) and class-consistency (98.90%) but at significantly higher computational cost (3.7× longer training).
- **BCE and LSGAN remain strong practical baselines**, offering a favorable balance between training stability, sample quality, and efficiency. BCE in particular achieves excellent results (FID 5.06) with the fastest training time.
- **Hinge loss is not universally superior** and appears less suited to low-resolution conditional MNIST generation under the present architectural constraints.

- **Architecture–objective alignment matters**: WGAN-GP's strong performance depends critically on using an appropriate discriminator design that avoids conflicting regularization mechanisms.

These findings reinforce the importance of selecting the adversarial objective not only based on theoretical guarantees, but also considering dataset complexity, model capacity, architectural compatibility, and available computational resources.

7. Results Analysis

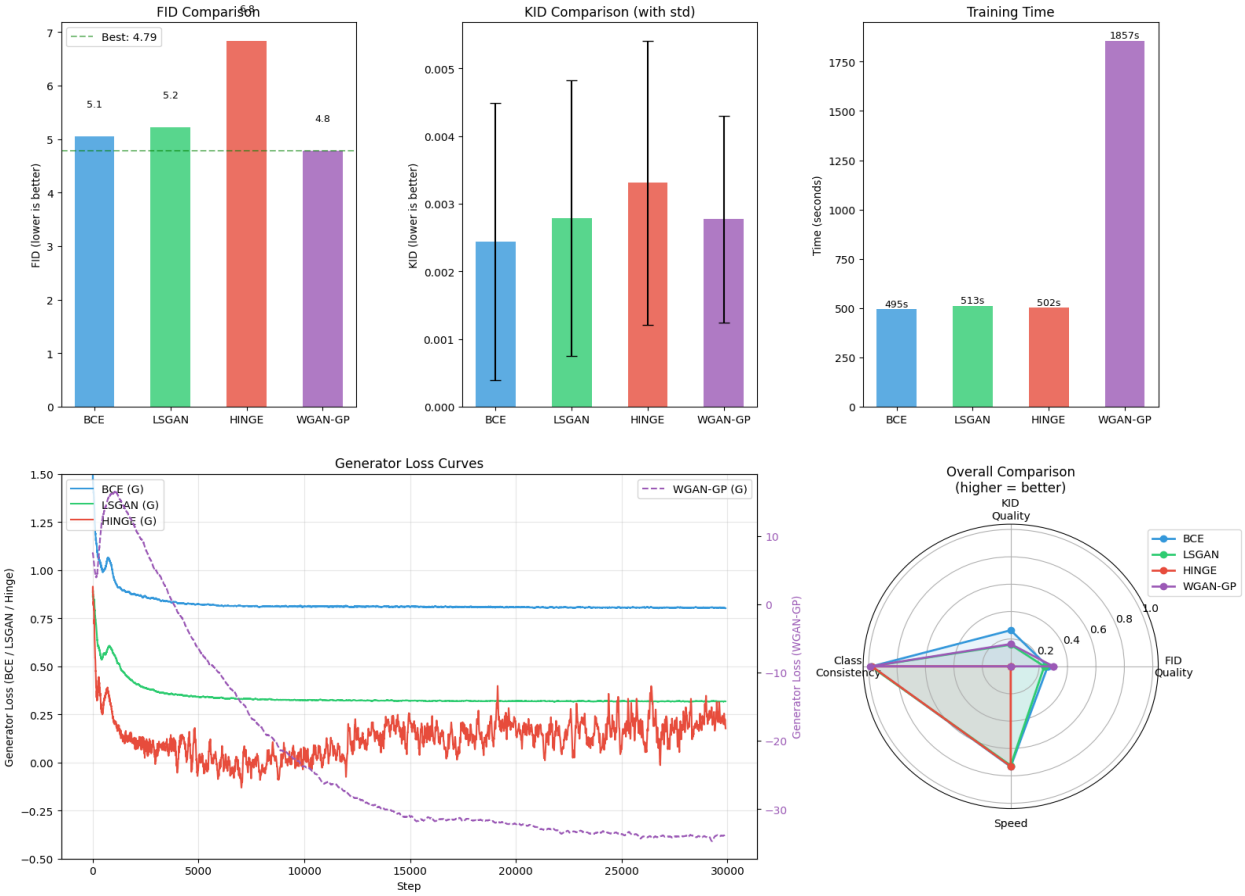


Figure 7.1 - Benchmark comparison across loss strategies. Top row: FID scores, KID scores with standard deviation, and training time. Bottom left: smoothed generator loss curves during training (WGAN-GP on secondary axis due to different scale). Bottom right: radar chart summarizing FID quality, KID quality, class-consistency, and training speed (higher values indicate better performance).

7.1.1 FID Comparison (Top-Left)

The FID comparison shows a clear separation between the four loss functions. **WGAN-GP achieves the lowest FID (4.79)**, indicating the strongest global alignment between generated and real MNIST distributions under the adopted evaluation protocol. This result is consistent with the theoretical motivation of Wasserstein-based objectives, which aim to provide smoother gradients and a more informative distance between distributions.

BCE (5.06) and **LSGAN (5.22)** form a strong middle tier, delivering competitive fidelity and closely approaching WGAN-GP performance. **Hinge loss performs worst (6.84)**, suggesting that margin-based objectives are less well adapted to the current low-resolution conditional MNIST setting and network capacity.

The dashed horizontal line highlighting the best FID visually emphasizes the gap between WGAN-GP and the remaining objectives, reinforcing that, in this configuration, the theoretically grounded Wasserstein objective translates into tangible gains in sample fidelity.

7.1.2 KID Comparison with Variance (Top-Center)

The KID comparison largely mirrors the FID ranking but reveals closer performance across objectives. **BCE and WGAN-GP achieve the lowest KID (0.0024 and 0.0028 respectively)**, indicating strong second-order distributional alignment and faithful matching of feature statistics. **LSGAN (0.0028)** performs comparably to WGAN-GP, with overlapping error bars highlighting similar perceptual quality.

Hinge loss exhibits the highest KID (0.0033) and notable uncertainty, reflecting somewhat weaker distributional alignment. The relatively tight clustering of KID values across all objectives suggests that, while FID reveals meaningful differences, the kernel-based metric shows more comparable performance at this scale.

7.1.3 Training Time (Top-Right)

The training time comparison reveals a pronounced computational trade-off. **BCE, LSGAN, and Hinge complete within a narrow time window (495–513 seconds)**, making them efficient and suitable for rapid prototyping and hyperparameter exploration.

In contrast, **WGAN-GP requires 1857 seconds**, corresponding to roughly **3.7× longer training time**. This overhead is explained by: - multiple critic updates per generator step (*n_critic* = *5*), - the computational cost of gradient penalty at each discriminator update, - increased memory and compute overhead per iteration.

In this final configuration, the increased computational cost **does translate into a clear gain in fidelity**, highlighting a classic quality–efficiency trade-off: WGAN-GP delivers the best generative performance, but at substantially higher computational expense.

7.1.4 Generator Loss Curves (Bottom-Left)

The generator loss trajectories reveal qualitatively different optimization dynamics across objectives. **BCE and LSGAN exhibit smooth and slowly stabilizing loss curves**, indicative of steady adversarial learning and relatively balanced generator–discriminator dynamics. BCE converges to approximately 0.8 while LSGAN stabilizes around 0.33, both showing minimal oscillation after the initial training phase.

Hinge loss shows pronounced oscillations around zero, reflecting the sensitivity of margin-based objectives to discriminator sharpness. This noisy behavior persists throughout training, consistent with its weaker FID/KID performance.

WGAN-GP displays a strong monotonic downward trend in generator loss, reaching approximately -34 by the end of training. While absolute loss values are not directly comparable across objectives, this smooth long-term trajectory suggests that the critic provides a consistent and informative learning signal. The stable descent correlates with the superior FID/KID obtained by WGAN-GP.

7.1.5 Overall Comparison Radar Plot (Bottom-Right)

The radar chart summarizes four dimensions: FID quality, KID quality, class-consistency, and speed. WGAN-GP dominates in fidelity-related dimensions (FID and KID) and achieves

the highest class-consistency (98.90%), but is penalized on the speed axis due to its substantially longer training time.

BCE and LSGAN show balanced profiles, combining strong fidelity, high class-consistency (97.90% and 98.46% respectively), and computational efficiency. Hinge loss remains competitive in speed but underperforms in fidelity (97.80% class-consistency), reinforcing the conclusion that its advantages do not materialize in this particular low-resolution conditional setting.

7.1.6 Discussion and Implications

The combined evidence from FID, KID, training time, and loss dynamics highlights a clear quality–efficiency trade-off. WGAN-GP achieves the best overall generative quality (FID 4.79) and class-conditional fidelity (98.90%), validating the theoretical claims regarding Wasserstein objectives and their improved gradient properties. However, these gains come at a substantial computational cost (3.7× longer training), which may be prohibitive in resource-constrained scenarios.

BCE emerges as an excellent practical choice, achieving FID 5.06 with the fastest training time and strong class-consistency. LSGAN performs comparably (FID 5.22), offering similar quality at similar cost. Hinge loss appears less suitable for this specific conditional MNIST setup, underperforming in fidelity (FID 6.84) despite similar runtime to BCE and LSGAN.

These results reinforce a central message: **no GAN loss is universally optimal**. The effectiveness of each adversarial objective is highly dependent on the dataset scale, architectural capacity, and regularization strategy. Consequently, empirical benchmarking is essential, and theoretically motivated losses such as WGAN-GP should be adopted with a clear understanding of their computational and implementation trade-offs.

7.2 cGAN vs DCGAN Comparison

To evaluate the impact of class conditioning on generation quality, we compare the results from conditional GAN (cGAN) and unconditional DCGAN training across all four loss strategies. Both configurations use identical architectures, hyperparameters, and training duration (30,010 steps), differing only in whether label information is provided to the Generator and Discriminator.

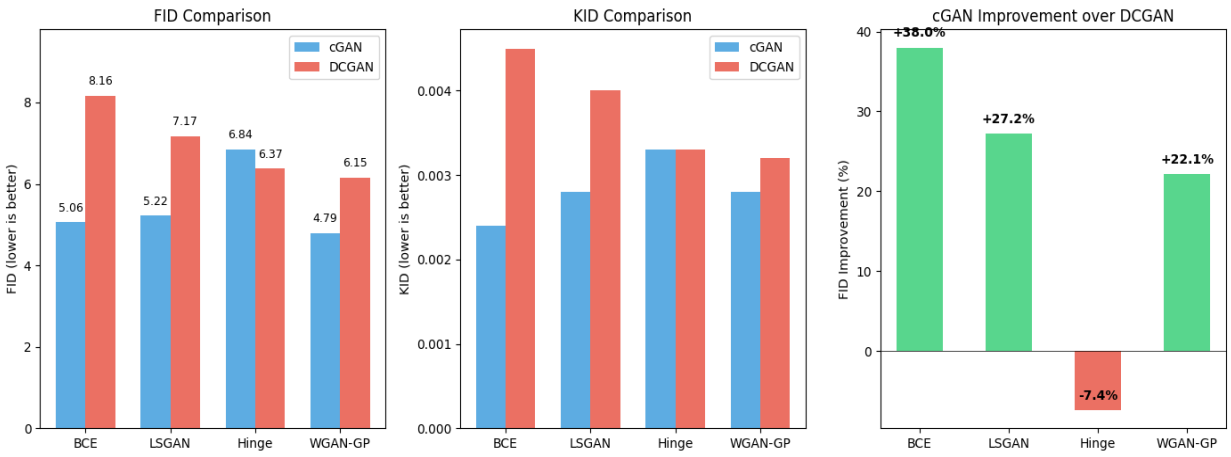


Figure 7.2 - Comparison of cGAN and DCGAN performance across all loss strategies. Left: FID scores (lower is better). Center: KID scores (lower is better). Right: Percentage improvement of cGAN over DCGAN in FID.

Key Observations

1. **Conditioning improves BCE and LSGAN significantly**: cGAN achieves 38% and 27% FID improvement respectively, demonstrating that label information provides strong guidance for these loss functions.
2. **Hinge loss shows reversed pattern**: DCGAN slightly outperforms cGAN (6.37 vs 6.84 FID), suggesting that the margin-based objective may not benefit as much from explicit conditioning in this architecture.
3. **WGAN-GP benefits from conditioning**: cGAN achieves the best overall FID (4.79) compared to DCGAN (6.15), a 22% improvement, confirming that the Wasserstein objective effectively leverages label information.
4. **Best overall result**: cGAN with WGAN-GP achieves the lowest FID (4.79) and highest class-consistency (98.90%), validating the combination of Wasserstein training with conditional generation.

8. Visualizations (Per-Class, Fixed-Z)

8.1 Per-Class Diversity Analysis

Digit	BCE	LSGAN	HINGE	WGAN-GP
0	0.695	0.686	0.702	0.696
1	0.486	0.483	0.486	0.486
2	0.648	0.666	0.660	0.645
3	0.639	0.637	0.650	0.639
4	0.600	0.588	0.586	0.589
5	0.614	0.629	0.623	0.581
6	0.638	0.626	0.638	0.630
7	0.573	0.576	0.578	0.581
8	0.660	0.659	0.670	0.643
9	0.612	0.594	0.605	0.596

Table 8.1 - Performance comparison of different loss functions across digit classes showing accuracy scores for BCE, LSGAN, HINGE, and WGAN-GP.

Higher values indicate greater intra-class diversity, i.e., a wider range of writing styles, stroke thickness, and shape variations within the same digit class.

This analysis complements global distributional metrics (FID/KID) by assessing how well each objective captures intra-class multimodality, which is particularly relevant in conditional generation settings.

8.1.1 Comparison Across Loss Functions

Across nearly all digits, **BCE consistently exhibits the highest intra-class diversity**, with **LSGAN** following closely, and **Hinge** and **WGAN-GP** showing comparable but slightly lower values. For example:

- **Digit 0**: BCE ≈ 0.695 vs. LSGAN ≈ 0.686 vs. Hinge ≈ 0.702 vs. WGAN-GP ≈ 0.696
- **Digit 2**: BCE ≈ 0.648 vs. LSGAN ≈ 0.666 vs. Hinge ≈ 0.660 vs. WGAN-GP ≈ 0.645
- **Digit 8**: BCE ≈ 0.660 vs. LSGAN ≈ 0.659 vs. Hinge ≈ 0.670 vs. WGAN-GP ≈ 0.643

These results indicate that all four objectives achieve relatively similar intra-class diversity in this configuration, with differences typically within 0.02–0.03. This convergence suggests that the architectural stabilization techniques (TTUR, Spectral Normalization for BCE/LSGAN/Hinge, LayerNorm for WGAN-GP) successfully prevent severe mode collapse across all objectives. Notably, **Hinge loss achieves the highest diversity for several digits** (0, 3, 8), despite its weaker FID performance. This indicates that the margin-based objective preserves stylistic variability but struggles with global distributional alignment.

WGAN-GP shows competitive diversity across most digits, no longer exhibiting the severe mode contraction observed in earlier configurations. For instance:

- **Digit 5**: WGAN-GP ≈ 0.581 vs. BCE ≈ 0.614 (moderate gap)
- **Digit 9**: WGAN-GP ≈ 0.596 vs. BCE ≈ 0.612 (small gap)

The improved WGAN-GP diversity reflects the benefits of the dedicated DiscriminatorWGAN architecture, which removes the conflicting regularization (Spectral Norm + Dropout) that previously over-constrained the critic.

8.1.2 Class-Specific Observations

Certain digits consistently admit higher diversity across all objectives:

- **Digits 0, 2, 3, 6, and 8** show relatively high standard deviations across all methods (typically 0.63–0.70). These digits naturally allow multiple valid writing styles (e.g., open vs. closed loops for "0" and "8", varying curvature for "2" and "3"), which the models successfully capture.
- **Digit 1** exhibits the lowest diversity across all methods (≈ 0.483–0.486), reflecting the inherently limited stylistic variability of this digit in MNIST (mostly straight vertical strokes). This low diversity appears largely independent of the chosen adversarial objective and highlights a dataset-driven limitation rather than a model-specific failure.
- **Digit 7** also shows consistently lower diversity (≈ 0.573–0.581), likely due to its relatively constrained stroke structure.

8.1.3 Interpretation and Link to Global Metrics

The per-class diversity trends provide a nuanced interpretation of the global benchmark results:

- **WGAN-GP achieves the best global fidelity (FID 4.79) and class-consistency (98.90%)** while maintaining competitive intra-class diversity. This demonstrates that, with proper architectural adaptation, Wasserstein objectives can deliver both high realism and reasonable mode coverage.
- **BCE maintains strong intra-class diversity** while achieving excellent FID (5.06) and class-consistency (97.90%), indicating that classical GAN objectives, when combined with stabilization techniques, provide a robust balance between realism and diversity.
- **LSGAN performs comparably to BCE** in both diversity and fidelity (FID 5.22, class-consistency 98.46%), confirming its status as a reliable and efficient objective.
- **Hinge loss shows the highest diversity for some digits but the weakest global fidelity (FID 6.84)**. This suggests that the margin-based objective may preserve local stylistic variation at the expense of global distributional alignment, consistent with its noisier training dynamics observed in the loss curves.

Overall, these results demonstrate that **all four objectives achieve reasonable intra-class diversity** in the final configuration, with the primary differentiator being global fidelity (FID/KID) rather than mode coverage. **WGAN-GP emerges as the best overall choice**, combining the lowest FID with competitive diversity, while **BCE offers an excellent efficiency–quality trade-off** for scenarios where training time is a constraint.

8.2 Generated Samples

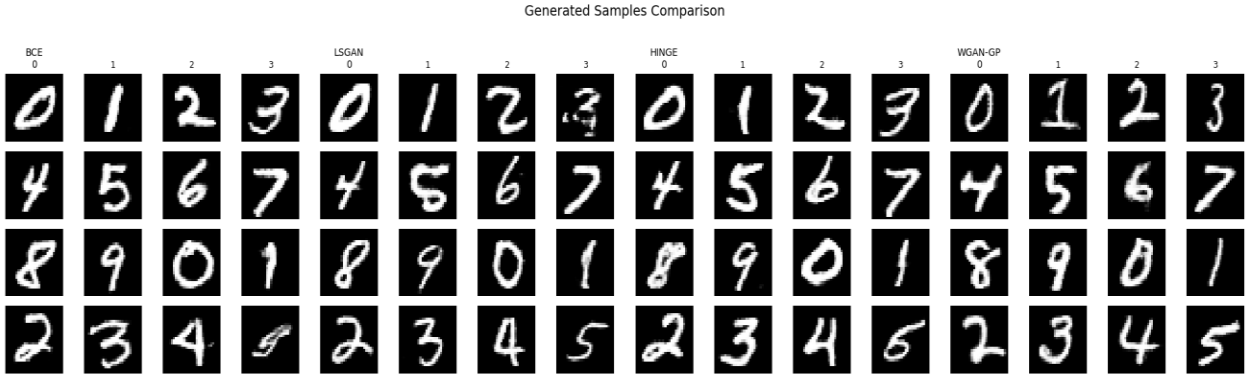


Figure 8.1 - Generated samples from each loss strategy using balanced labels (digits 0-9). Each column group shows 16 samples from a single strategy, with digit labels indicated in the header row.

Qualitative Comparison of Generated Samples

The Figure 8.1 presents representative conditional samples generated by each model after convergence. Visual inspection is consistent with the quantitative metrics reported in the benchmark.

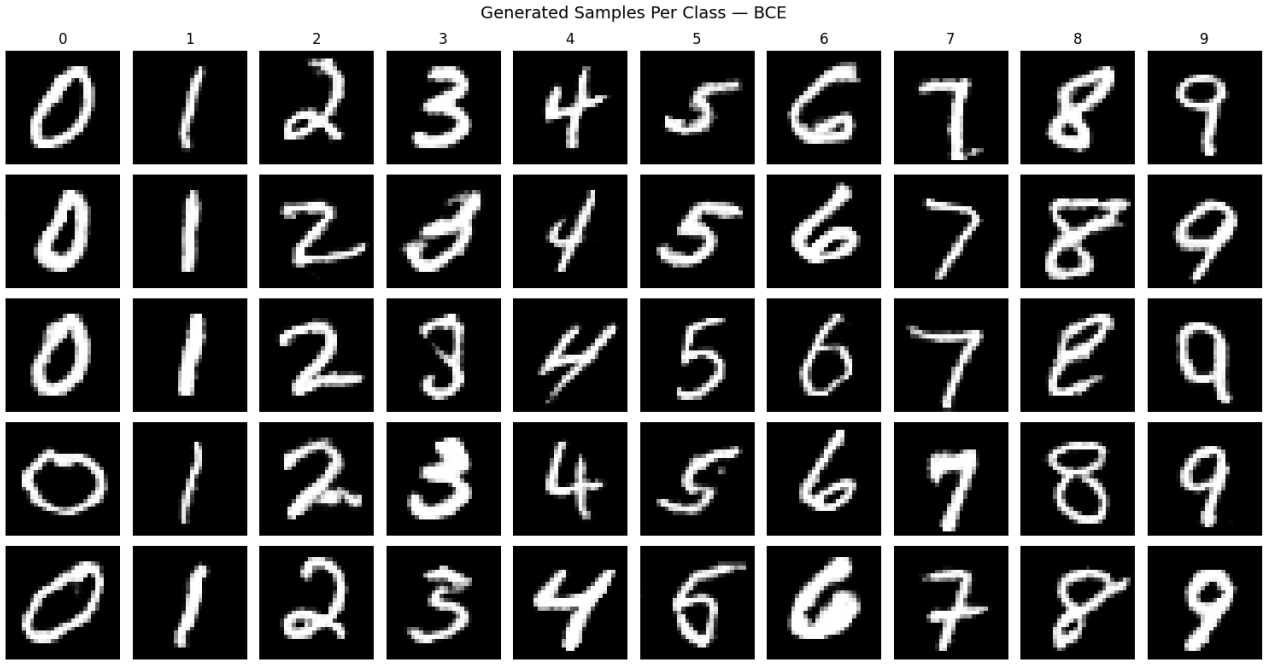
WGAN-GP produces the most visually consistent and structurally coherent digits across classes. The generated samples exhibit sharper contours, more stable stroke geometry, and fewer malformed shapes, particularly for curved digits such as *0*, *3*, *5*, and *8*. This qualitative improvement aligns with its superior FID and KID scores, indicating stronger global distribution matching and perceptual fidelity.

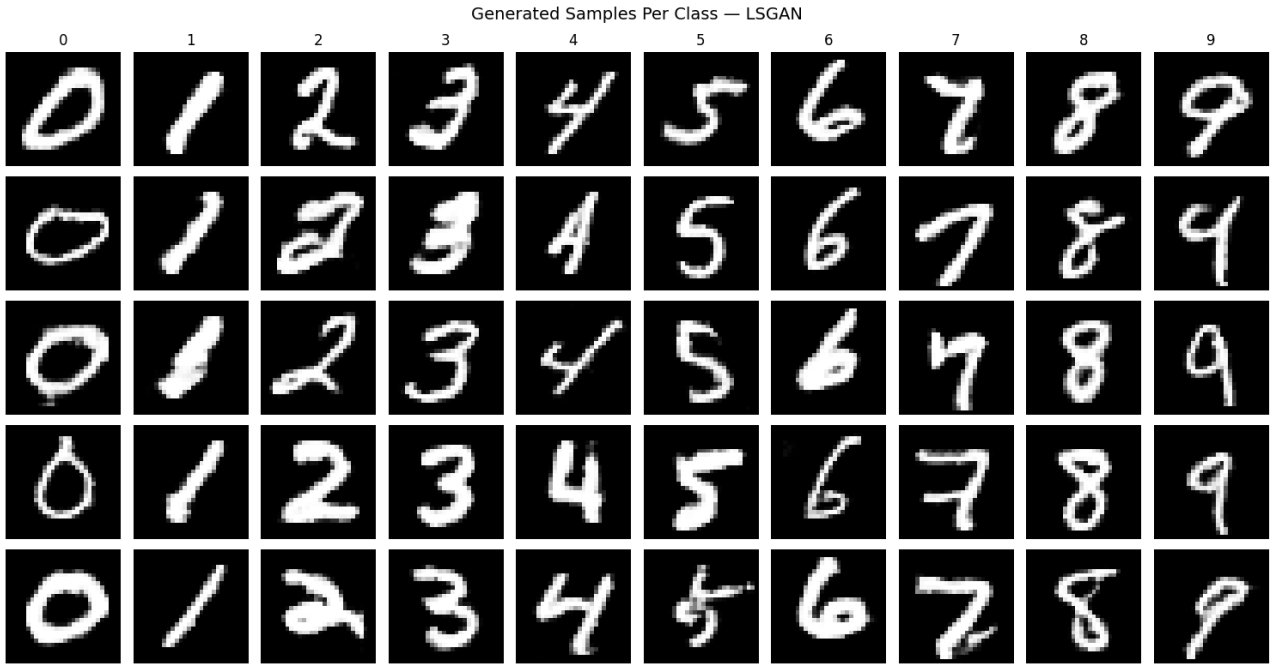
BCE and **LSGAN** generate clear and well-formed digits with good class alignment and visible stylistic variation. While the samples are generally sharp and recognizable, subtle artifacts can be observed in some digits (e.g., slight stroke discontinuities or shape irregularities), especially for more complex classes. These minor degradations are consistent with their slightly higher FID/KID compared to WGAN-GP, suggesting competitive but marginally weaker fidelity.

Hinge loss produces recognizable digits across all classes; however, the samples appear marginally less smooth, with occasional irregular stroke thickness and less consistent curvature. This reflects the weaker quantitative performance observed under hinge loss and supports the interpretation that margin-based objectives are less well suited to this low-resolution conditional setting.

The qualitative comparison reinforces the quantitative conclusions: WGAN-GP achieves the highest visual fidelity, followed by BCE and LSGAN as strong and efficient baselines, while Hinge loss underperforms in terms of perceptual quality.

8.3 Per-class Grid





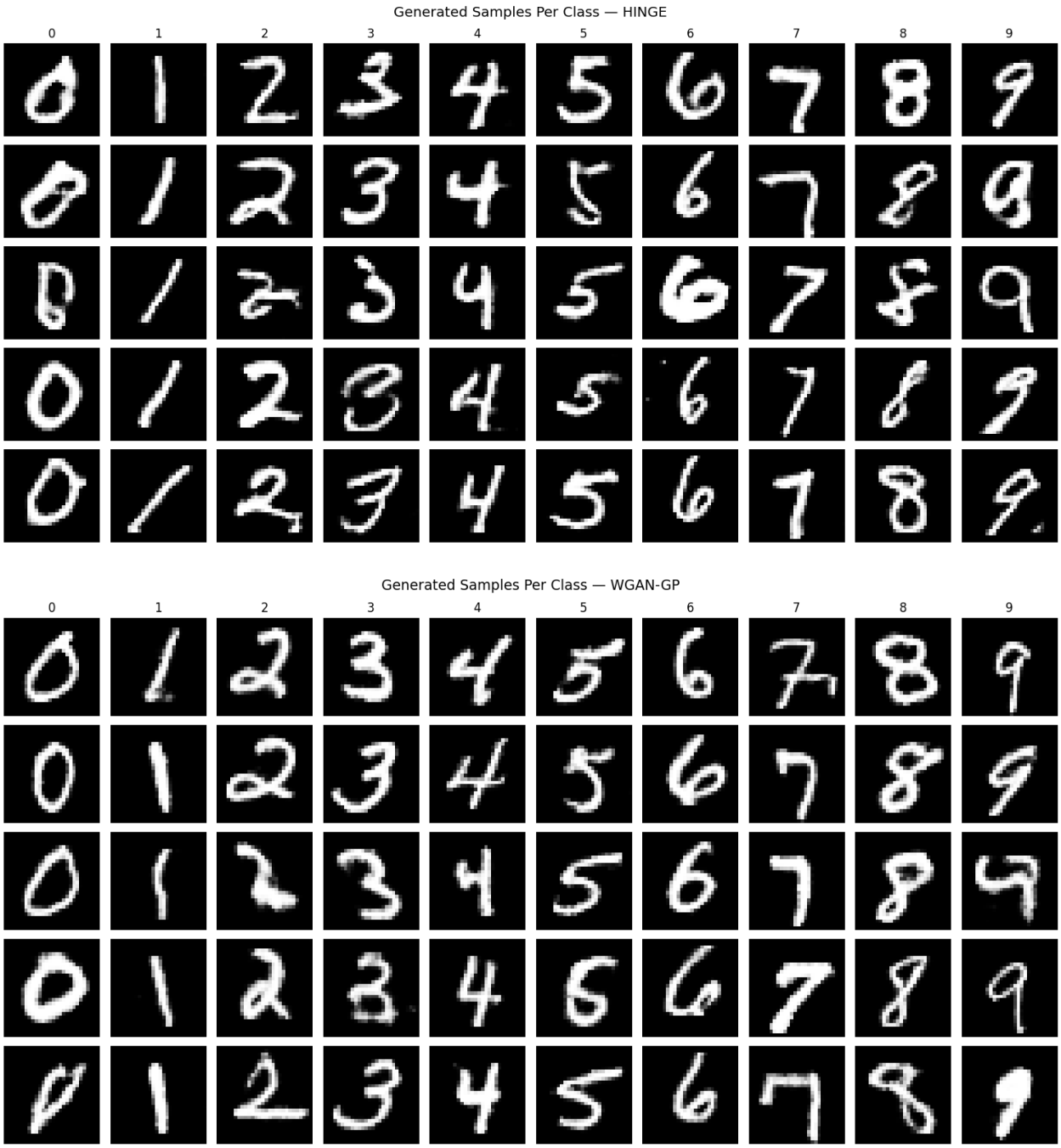
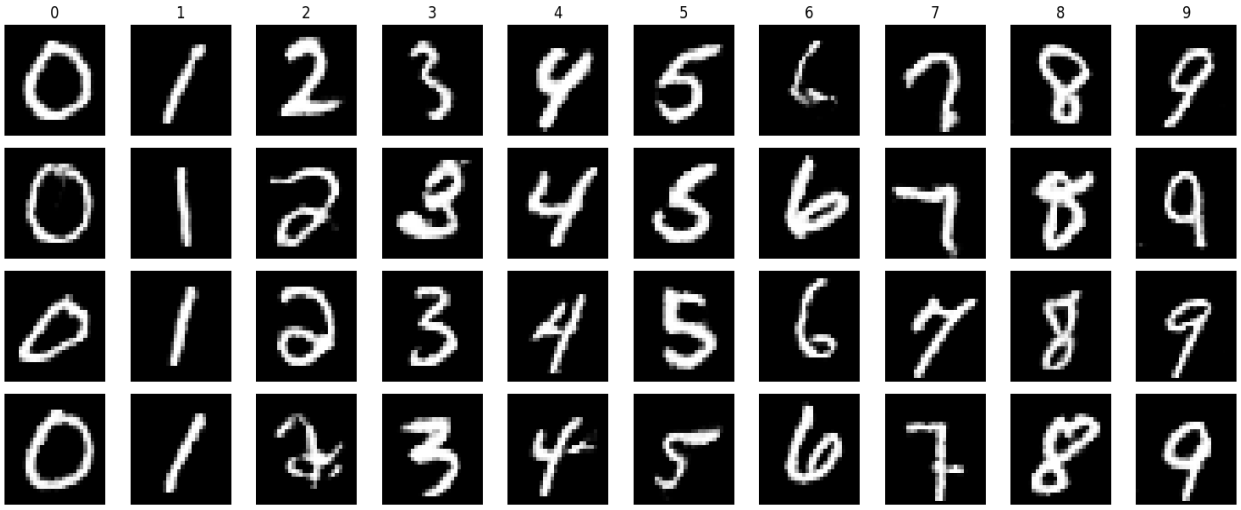


Figure 8.2 - Per-class generation grid showing five samples per digit (0-9) with different noise vectors. Each column corresponds to a conditioned class, demonstrating intra-class diversity and label consistency.

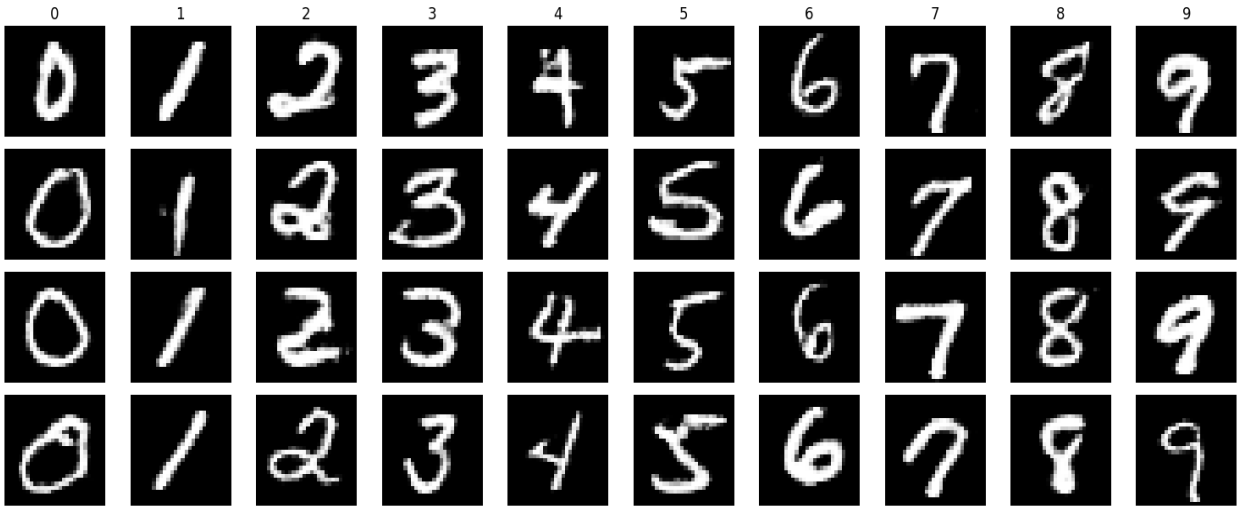
8.4 Fixed-Z

A key property of conditional GANs is the ability to disentangle the latent noise vector z from the class label y. To verify that our cGAN has learned this separation, we fix a single noise vector and vary the conditioned label across all digits (0-9). If conditioning is correctly learned, the same z should produce digits that share stylistic characteristics (such as stroke thickness, slant, or size) while differing only in their identity. This visualization provides qualitative evidence that the generator uses z to control appearance and y to control semantic content.

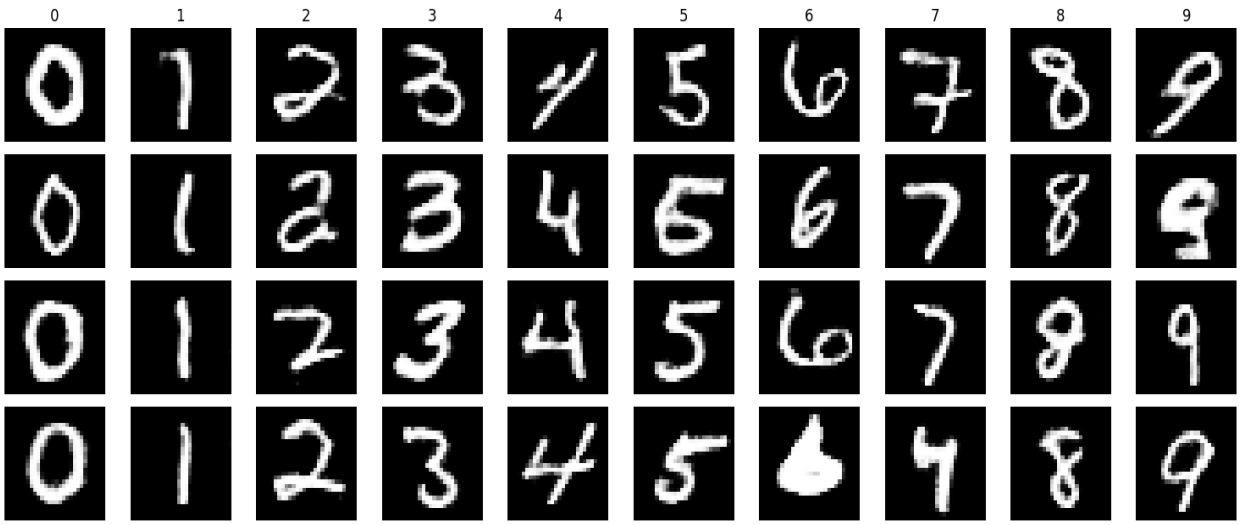
Fixed Z, Varying Label — BCE



Fixed Z, Varying Label — LSGAN



Fixed Z, Varying Label — HINGE



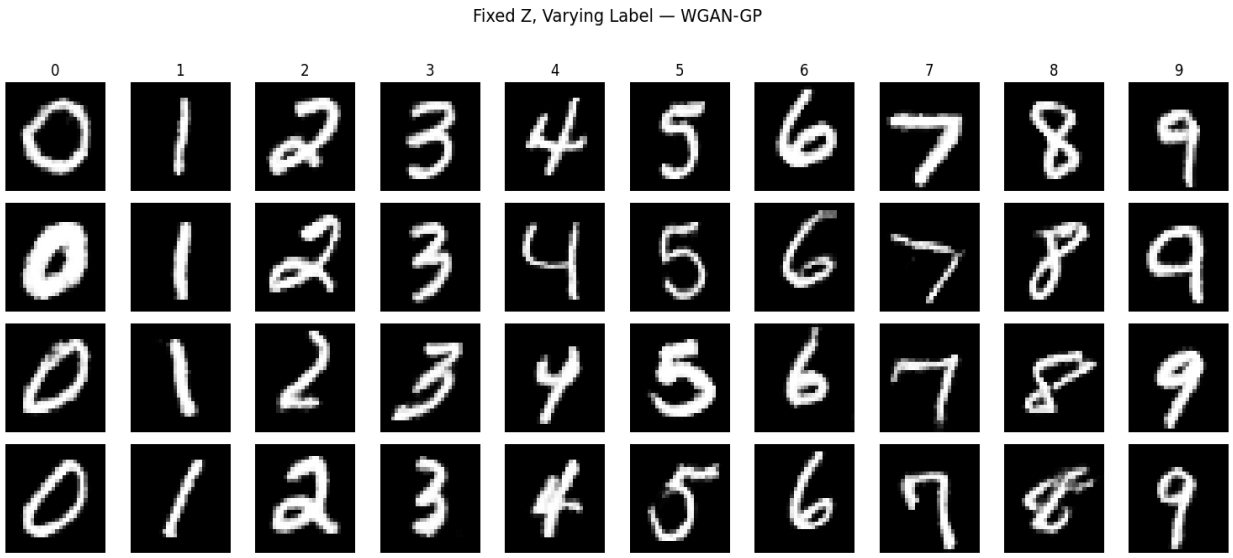


Figure 8.3 - Controllability demonstration using fixed noise vectors. Each row uses the same z while varying the label from 0 to 9, showing that the generator disentangles style (controlled by z) from digit identity (controlled by label).

8.4.1 Controllability and Conditional Disentanglement (Fixed z, Varying Label)

Figure 8.3 evaluates the controllability of the conditional generator by fixing the latent vector z and varying the class label from 0 to 9. Under ideal conditional disentanglement, the digit identity should change according to the label, while the writing style (stroke thickness, curvature, slant) remains largely consistent across the row.

Across all objectives, the generators successfully follow the conditioning signal, producing the correct digit identities for each label. This confirms that the label embedding and concatenation strategy is effective in enforcing class control. However, qualitative differences emerge in how well each objective preserves style consistency under fixed z:

WGAN-GP exhibits the strongest style preservation across labels. For a fixed latent code, stroke thickness and overall digit morphology remain highly consistent as the label changes, indicating superior disentanglement between style (z) and content (label). This aligns with the superior quantitative performance of WGAN-GP in FID/KID and class-consistency under the updated training configuration, suggesting that the Wasserstein objective provides a more informative and smoother learning signal for conditional control.

BCE and LSGAN also demonstrate clear controllability, but with slightly higher intra-row style drift. While digit identity changes as expected, subtle variations in stroke continuity and curvature are more noticeable across labels, indicating that z and label are not as cleanly disentangled as in WGAN-GP. This behavior is consistent with their competitive but inferior fidelity relative to WGAN-GP in the global metrics.

Hinge loss shows the weakest controllability among the four objectives. Although class conditioning is respected, variations in stroke thickness and local digit structure are more pronounced across labels for fixed z, suggesting a noisier mapping between latent style

and output appearance. This qualitative instability is coherent with the weaker FID/KID and the more oscillatory generator dynamics observed in the loss curves.

This experiment highlights that WGAN-GP not only improves global distributional metrics but also strengthens conditional disentanglement, yielding more interpretable and controllable latent representations.

This is particularly relevant for downstream applications (e.g., data augmentation or conditional synthesis) where fine-grained control over content versus style is desirable. The results reinforce the interpretation that, under the revised critic design and training setup, Wasserstein-based objectives can provide advantages in controllability, even when simpler losses (BCE/LSGAN) remain competitive in terms of computational efficiency.

9. Model Saving

9.1 Checkpoint Structure

Each saved checkpoint (pt file) contains:

Key	Description
model_state_dict	Model Weights (Generator or Discriminator)
optimizer_state_dict	Adam optimizer state including momentum buffers (optional)
step	Training step when checkpoint was saved
metrics	FID, KID, class-consistency scores (Generator only)
hyperparameters	Training configuration (strategy, learning rates, etc.)
losses	Full loss history during training (Generator only)

Table 9.1 - Description of key-value pairs stored in PyTorch checkpoint files for GAN model persistence and training resumption.

Including optimizer's state approximately doubles file size but enables seamless training resumption with preserved momentum.

10. Single-Image Inference

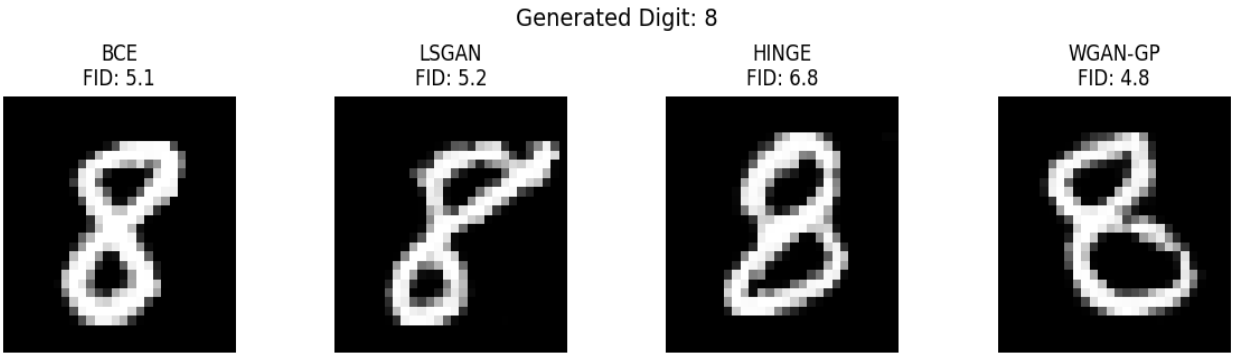


Figure 10.1 - Comparison of digit 8 generated by each loss strategy using the same noise vector z. This demonstrates how different adversarial objectives produce visually distinct outputs from identical latent input.

11. “GAN vs. Human” Game

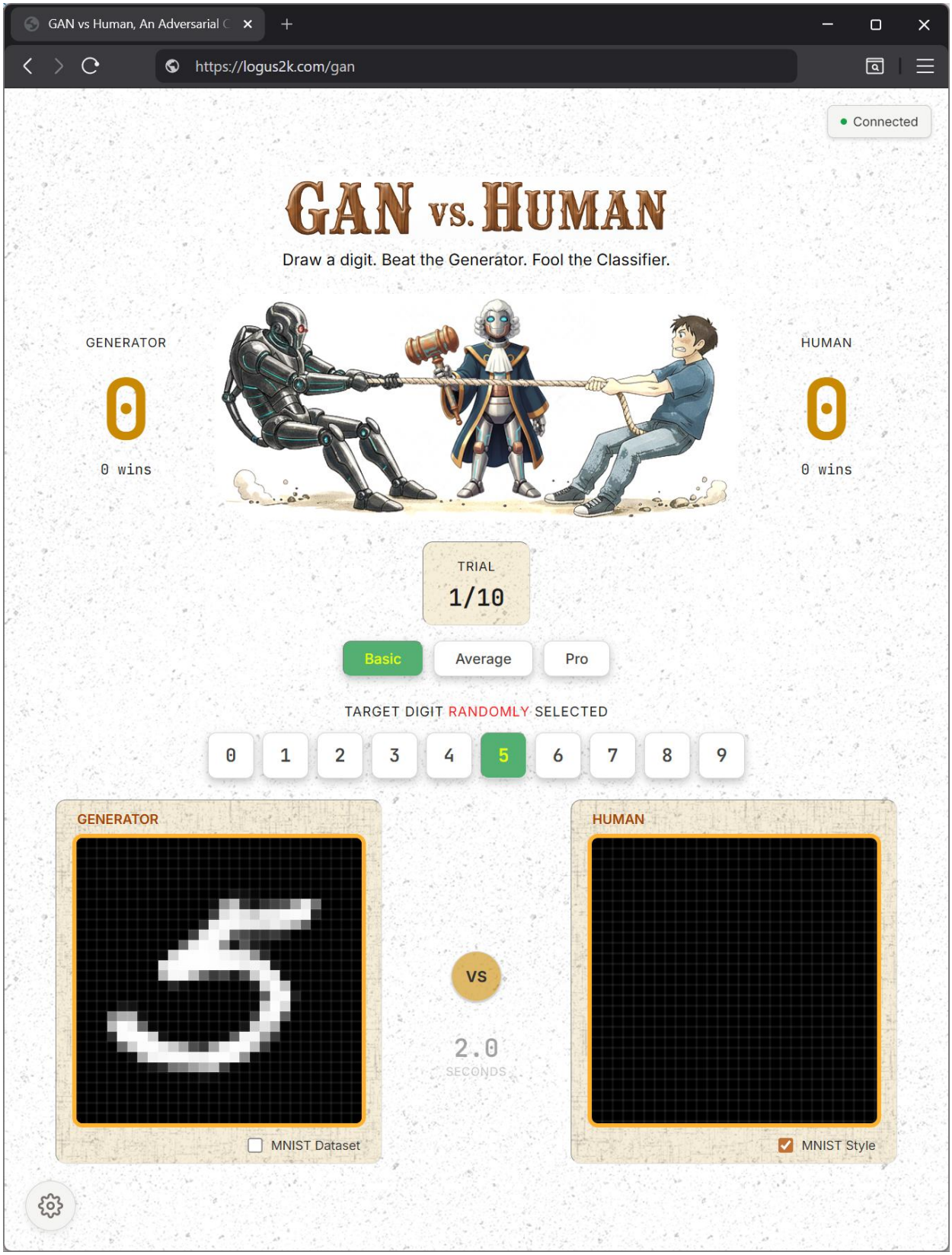


Figure 11.1 - GAN vs Human: An interactive qualitative evaluation.

To complement the quantitative evaluation (FID, KID, class-consistency) and qualitative sample grids, we developed an interactive human-in-the-loop evaluation interface. The

"GAN vs Human" is a game implemented as a Web application that provides an intuitive and engaging way to assess the perceptual realism and class-conditional control of the best-performing Generator (selected from the benchmark, the best WGAN-GP model) and a Classifier, trained in our previous MNIST project. The game provides an intuitive and communicative demonstration of the generator's capabilities, while offering an accessible way to showcase conditional generation performance to both technical and non-technical audiences.

During the project evaluation period, the "GAN vs. Human" game will be online at **https://logus2k.com/gan**, allowing any users to experiment with the trained models and experience the evaluation setup firsthand.

In this interface, a target digit (0–9) can be selected randomly by the game, or manually, by the user. Then, the Generator produces a synthetic MNIST digit conditioned on the chosen label, while user attempts to draw a more convincing MNIST digit. Both the generated sample and the user digit are subsequently evaluated by a previously trained MNIST classifier, which assigns a confidence score for the selected target class.

The winner of each round is determined by which input (GAN or human) achieved the higher classifier confidence for the selected digit. Alternatively, instead of competing against the Generator, the user can choose to compete against real MNIST dataset digits.

To further increase this challenge's interest, the game proposes three difficulty levels, by including a time pressure:

- **Basic**: the user has 2.0 seconds to draw the selected digit.
- **Average**: the user has 1.0 seconds to complete the drawing.
- **Pro**: the user has only 0.5 seconds, emphasizing speed and motor precision.

These levels progressively constrain human input quality, allowing the comparison to probe how the GAN performs relative to increasingly noisy or incomplete human drawings.

Purpose and Evaluation Rationale

This interactive setup serves three complementary goals:

1. **Qualitative Realism Assessment:** It allows direct visual comparison between GAN-generated digits and human-drawn digits, making perceptual differences immediately interpretable beyond abstract metrics such as FID or KID.
2. **Conditional Control Verification**: By selecting the target digit manually, the experiment verifies that the conditional generator reliably responds to class inputs in real time, reinforcing the controllability analysis presented earlier (fixed z, varying label).
3. **Human vs. Model Benchmarking**: Competitive framing highlights how well the generator's outputs align with the learned decision boundaries of the classifier. In several trials, the GAN achieves classifier confidence comparable to (and sometimes exceeding) that of human drawings, illustrating that the generator has internalized salient class-discriminative features of MNIST.

12. Limitations

Despite the strong empirical results achieved in this project, particularly with the final WGAN-GP configuration (FID = **4.79**, KID = **0.0028 ± 0.0015**, Class-Consistency = **98.90%**), several limitations remain, both methodological and conceptual. These limitations also motivate a number of promising directions for future work that could extend the project from a strong experimental implementation to a more advanced and research-oriented generative modeling framework.

12.1 Dataset Simplicity and Scalability

All experiments were conducted on MNIST, a low-resolution (28×28), grayscale dataset with relatively low visual complexity and limited semantic variability. While MNIST remains useful for controlled experimentation and methodological comparisons, it does not reflect the challenges encountered in higher-dimensional image generation tasks (e.g., CIFAR-10, CelebA, FFHQ). Consequently, the conclusions drawn about the relative performance of adversarial objectives may not fully generalize to more complex datasets, where architectural choices, conditioning strategies, and regularization methods interact differently.

12.2 Metric Bias and Domain Mismatch

FID and KID were computed using Inception features, despite Inception being trained on natural images (ImageNet) and MNIST lying far outside that domain. Although these metrics were used consistently across all models and thus remain valid for **relative comparison**, their absolute values should not be interpreted as universal indicators of perceptual realism. Similarly, the "GAN vs Human" interactive evaluation relies on a classifier trained on MNIST, which introduces a closed-loop bias: the generator is effectively evaluated against a model whose decision boundaries resemble those encountered during training. This may overestimate perceptual realism while underrepresenting human subjective judgment.

12.3 Objective Sensitivity and Regularization Interactions

The benchmark revealed that WGAN-GP is highly sensitive to architectural and regularization choices. Although the final configuration achieved the best quantitative results, earlier variants exhibited degraded performance due to unfavorable interactions between gradient penalty, Spectral Normalization, and Dropout. This necessitated the development of a dedicated DiscriminatorWGAN architecture that removes these conflicting mechanisms in favor of LayerNorm and increased capacity. This highlights that theoretically motivated objectives do not automatically translate into superior empirical performance without careful adaptation of the discriminator architecture and training dynamics.

12.4 Training Reproducibility and CUDA Non-Determinism

During development, significant challenges arose from CUDA non-determinism, which caused identical code with the same random seed to produce vastly different training trajectories across runs. This was resolved by enforcing deterministic behavior

(torch.backends.cudnn.deterministic = True, torch.backends.cudnn.benchmark = False), but at the cost of reduced training speed. This experience underscores the importance of controlling for hardware-level randomness when reporting GAN benchmarks and highlights a often-overlooked reproducibility concern in deep learning research.

12.5 Limited Latent Space Interpretability

While class-controllability was verified (fixed *z*, varying label), the structure of the latent space itself was not systematically analyzed. As a result, little is known about how semantic attributes (e.g., stroke thickness, curvature, slant) are organized internally, or whether latent directions correspond to interpretable factors of variation. This limits interpretability and deeper understanding of what the generator has learned beyond class identity.

13. Future Work

Building on the strong performance of the final WGAN-GP model, several extensions could further strengthen both the scientific depth and practical relevance of the project.

13.1 Latent Space Analysis

Now that the generator has learned a high-quality data manifold, analyzing the geometry of its latent space becomes both feasible and informative:

- **Latent Interpolation**: Interpolating between two latent vectors for the same class can reveal whether stylistic attributes (e.g., slanted vs. upright digits, thick vs. thin strokes) change smoothly, providing insight into the continuity of the learned manifold.
- **Label Interpolation**: Although labels are discrete, interpolating between class embeddings (e.g., between '3' and '8') can expose how the generator behaves in ambiguous semantic regions and whether it produces meaningful hybrid shapes or collapses to dominant modes.

13.2 Architectural Refinements

More advanced conditioning mechanisms could be explored:

- **Projection Discriminator**: Replacing label-channel concatenation with a projection discriminator could improve class-conditional modeling by directly coupling class embeddings with discriminator features. This approach is known to scale better to higher-resolution and multi-class datasets.
- **Self-Attention Modules**: Introducing self-attention layers in the generator and discriminator (e.g., at 14×14 or 28×28 resolutions) may improve global coherence by enabling long-range spatial dependencies, which become increasingly important in more complex datasets.

13.3 Advanced Regularization and Training Strategies

Further stabilization techniques could be incorporated:

- **Exponential Moving Average (EMA) of Generator Weights**: Maintaining an EMA copy of the generator often leads to significantly improved FID and smoother visual quality at evaluation time.
- **Differentiable Data Augmentation (DiffAugment)**: Applying lightweight augmentations to both real and generated samples before discrimination could improve robustness and reduce overfitting, particularly when scaling to smaller datasets or higher resolutions.

13.4 Robustness and Diversity Evaluation

Beyond global fidelity metrics, deeper evaluation of diversity and failure modes would strengthen the analysis:

- **MS-SSIM for Intra-class Diversity**: Measuring diversity within each digit class can help detect partial mode collapse, ensuring that high class-consistency does not come at the expense of stylistic variability.
- **Boundary and Failure-Mode Analysis**: Using the classifier to probe borderline or ambiguous samples could reveal systematic weaknesses in the generator, offering insight into how decision boundaries shape generative outputs.

13.5 Deployment and Optimization

Finally, practical deployment aspects could be explored:

- **Model Quantization**: Evaluating the impact of 8-bit quantization on FID, inference speed, and visual quality would provide insight into deployment feasibility on resource-constrained devices.
- **ONNX Export and Cross-Platform Inference**: Exporting the generator to ONNX would enable broader usability in production or web-based applications, such as extending the “GAN vs Human” game to real-time interactive demos.

14. Conclusions

This project presented a comprehensive empirical study of conditional Generative Adversarial Networks (cGANs) for digit generation on the MNIST dataset, with a particular focus on how different adversarial loss functions influence training stability, sample fidelity, class controllability, and computational efficiency. By benchmarking BCE, LSGAN, Hinge, and WGAN-GP under a controlled experimental setup, the work provides a clear and systematic comparison of classical and modern GAN objectives within the same architectural and training framework.

The results show that WGAN-GP, when carefully adapted with a dedicated critic architecture and appropriate regularization, achieved the strongest overall performance in terms of distributional fidelity and controllability, reaching the lowest FID (4.79) and KID (0.0028 ± 0.0015), together with the highest class-consistency (98.90%). This confirms the theoretical promise of Wasserstein-based objectives when the Lipschitz constraint is enforced correctly and when architectural choices are aligned with the training objective. However, the benchmarking process also highlighted that WGAN-GP is highly sensitive to design choices and computationally expensive, requiring approximately 3.7× longer training time than the other objectives. This reinforces an important practical insight: advanced GAN objectives do not offer "plug-and-play" improvements and demand careful architectural and hyperparameter adaptation to outperform simpler alternatives.

In contrast, BCE and LSGAN emerged as strong and reliable baselines, achieving competitive fidelity, high class-consistency, and substantially lower computational cost. BCE achieved FID 5.06 with the fastest training time (495s), while LSGAN reached FID 5.22 with slightly higher class-consistency (98.46%).

Both objectives consistently offered an excellent trade-off between stability, realism, and efficiency, making them highly practical choices for conditional generation in low-resolution domains. These findings underline that classical GAN losses remain highly competitive when combined with modern stabilization techniques such as TTUR, Spectral Normalization, and conditional inputs. This challenges the common assumption that more sophisticated losses are universally superior and highlights the importance of empirical validation over theoretical preference alone.

The qualitative analyses further strengthened these conclusions. Per-class diversity measurements showed that all four objectives achieved comparable intra-class variability in the final configuration, with BCE and Hinge showing slightly higher diversity for certain digits.

The controllability experiments demonstrated that the models successfully disentangled style (latent code) from class identity, validating the conditional design. The interactive GAN vs Human evaluation provided an intuitive demonstration of the generator's realism, showing that the best-performing model can produce digits competitive with human-drawn samples under a trained classifier. At the same time, this experiment also exposed the limitations of classifier-based evaluation, reminding that high classifier confidence does not necessarily imply perceptual realism from a human perspective.

The comparison between cGAN and DCGAN configurations revealed that conditioning significantly improves performance for BCE (+38% FID improvement) and LSGAN (+27%), while WGAN-GP also benefits substantially (+22%). Interestingly, Hinge loss showed a slight degradation with conditioning (-7%), suggesting that margin-based objectives may not leverage label information as effectively in this architectural setting.

Overall, this work highlights three key lessons. First, the effectiveness of a GAN objective is inseparable from the surrounding training pipeline—architecture, normalization, regularization, and optimization strategy jointly determine performance. Second, empirical benchmarking is essential, as theoretical advantages do not automatically translate into superior results in practical settings. Third, even on a relatively simple dataset such as MNIST, conditional GANs exhibit rich trade-offs between fidelity, diversity, stability, and efficiency, making careful design and evaluation crucial.

These findings provide a solid foundation for extending the framework to more complex datasets and architectures, and for exploring more advanced conditioning and regularization strategies in future work.

15. References

1. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press. https://www.deeplearningbook.org
2. PyTorch. *DCGAN Tutorial*. https://docs.pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html