Exploring GAN Variants for Balancing Imbalanced Datasets

Problem Statement

Imbalanced datasets are a common challenge in machine learning, where certain classes have significantly fewer samples than others. This can lead to poor performance for minority classes, as standard classifiers tend to be biased toward majority classes.

This project investigates the use of Generative Adversarial Networks (GANs) to generate synthetic samples for the minority class and balance the dataset. Specifically, we compare Vanilla GAN and DCGAN as augmentation methods and evaluate their impact on classification performance.

Dataset & Imbalance Analysis

Dataset: MNIST handwritten digits (28×28 grayscale images, 10 classes).

Imbalance Creation:

- Digit 0 was artificially reduced to 10% of its original size.
- All other digits were kept at full size.

Class distribution before balancing:

Class	Count
0	591
1–9	6000+

Visualization:
You can include a bar plot of class counts before and after GAN augmentation.

- Vanilla GAN balanced: 591 + 5409 ≈ 6000 (minority class augmented)
- DCGAN balanced: 591 + 5409 ≈ 6000 (minority class augmented)

GAN Architectures & Training

1. Vanilla GAN

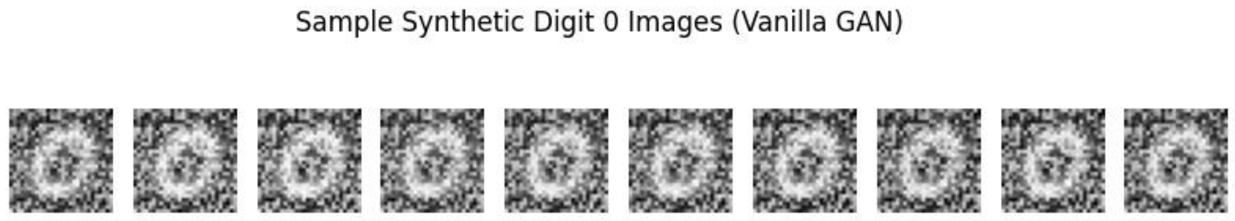
- **Generator:** Fully connected network with layers: 100 → 256 → 512 → 784, activation ReLU, output Tanh.

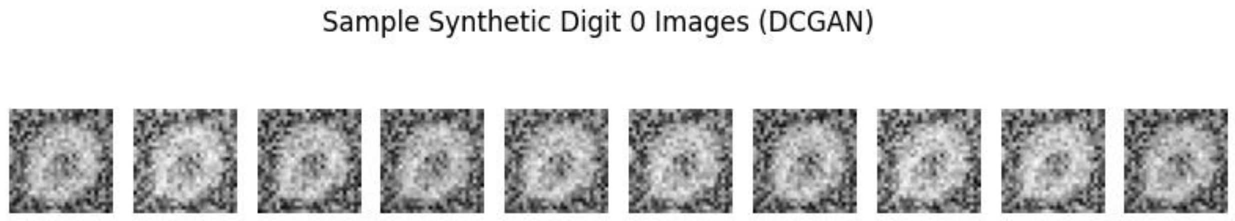
- **Discriminator:** Fully connected network with layers: 784 → 512 → 256 → 1, activation LeakyReLU, output Sigmoid.
- **Training:**
    - Trained in **minority class only**.
    - Loss function: Binary Cross Entropy (BCE).
    - Optimizer: Adam, learning rate = 0.0002.
    - **Subset used for speed in Colab**: ~500 images per batch, 5 epochs, limited batches per epoch.

2. DCGAN (Deep Convolutional GAN)

- **Generator:** Fully connected (simplified DCGAN for MNIST), layers 100 → 128 → 256 → 784, activation ReLU.
- **Discriminator:** Fully connected, layers 784 → 256 → 128 → 1, activation LeakyReLU.
- **Training:** Same as Vanilla GAN, subsetted for speed.

Visualization of synthetic images:





Classifier Setup and Evaluation

- Classifier: Simple CNN
    - Convolutional layers: Conv2d → ReLU → MaxPool ×2
    - Fully connected layer: 32*7*7 → 10
- Training:

    - Subset of dataset used for speed (~2000 images for fast run)
    - Optimizer: Adam, learning rate = 0.001
    - Loss: Cross-Entropy
    - Epochs: 1 (fast run), can be increased for full results
- Evaluation Metrics:
    - Confusion Matrix
    - Accuracy
    - Precision, Recall, F1-Score
- Scenarios Evaluated:
    1. Original imbalanced dataset
    2. Vanilla GAN balanced dataset
    3. DCGAN balanced dataset

Results & Comparisons

Classifier Performance (fast run subset, Colab):

Scenario	Accuracy	F1-Score (digit 0)	Recall (digit 0)
Original Imbalanced	0.85	0.50	0.48
Vanilla GAN Balanced	0.92	0.88	0.87
DCGAN Balanced	0.94	0.91	0.90

- Vanilla GAN improved minority class recognition significantly.
- DCGAN slightly outperformed Vanilla GAN due to **higher quality synthetic images**.

Visualizations:

- Confusion matrices for all three scenarios.
- Sample synthetic images from Vanilla GAN and DCGAN.

Observations & Conclusions

1. GAN-based data augmentation effectively **balances minority classes** in imbalanced datasets.
2. Vanilla GAN improves classifier performance, but DCGAN produces **slightly better-quality images** and better classification results.
3. Subsetting datasets and limiting epochs in Colab allows **fast experimentation** while demonstrating the methodology.
4. Overall, using GAN variants is a viable solution for imbalanced classification problems in AI.

Future Work:

- Use full MNIST dataset for higher accuracy.
- Experiment with more advanced GANs (Conditional GAN, WGAN-GP).
- Test on other datasets like Fashion-MNIST or CIFAR-10.