Special Topics in AI, Section 1, Spring 2024/2025

Dr. Yousef Sanjalawe

KASIT, JU

Exploring GAN Variants for Balancing Imbalanced Datasets

Problem Statement

Imbalanced datasets present a problem in machine learning, specially in classification tasks, where models often become biased towards the majority class, which give us poor performance in detecting minority class instances. In the dataset I used (COVID-19 image), this imbalance can result in the under-detection of positive cases, which is important to avoid.

This project addresses the imbalance problem by using Generative Adversarial Networks (GANs) to synthetically generate samples for the underrepresented classes. The generated synthetic data will be used to balance the minority class, and the impact of this augmentation will be measured using standard classification performance.

Description of Dataset & Imbalance Analysis

The dataset used in this project, sourced from the Hugging Face repository (yuighj123/covid-19-classification), contain two columns , consists of medical images categorized into three classes.

```
Dataset splits: DatasetDict({
    train: Dataset({
        features: ['image', 'label'],
        num_rows: 251
    })
    test: Dataset({
        features: ['image', 'label'],
        num_rows: 66
    })
})
```

The images are labeled as "Covid" or "Normal", or "Viral Pneumonia". The dataset is imbalanced, and the Covid class is overrepresented by 41 compared to the other two classes.

This class imbalance can lead to machine learning algorithms being biased towards the majority class, limiting their ability to correctly identify instances of the minority class. This is especially relevant in medical diagnosis tasks where misclassification has negative consequences.

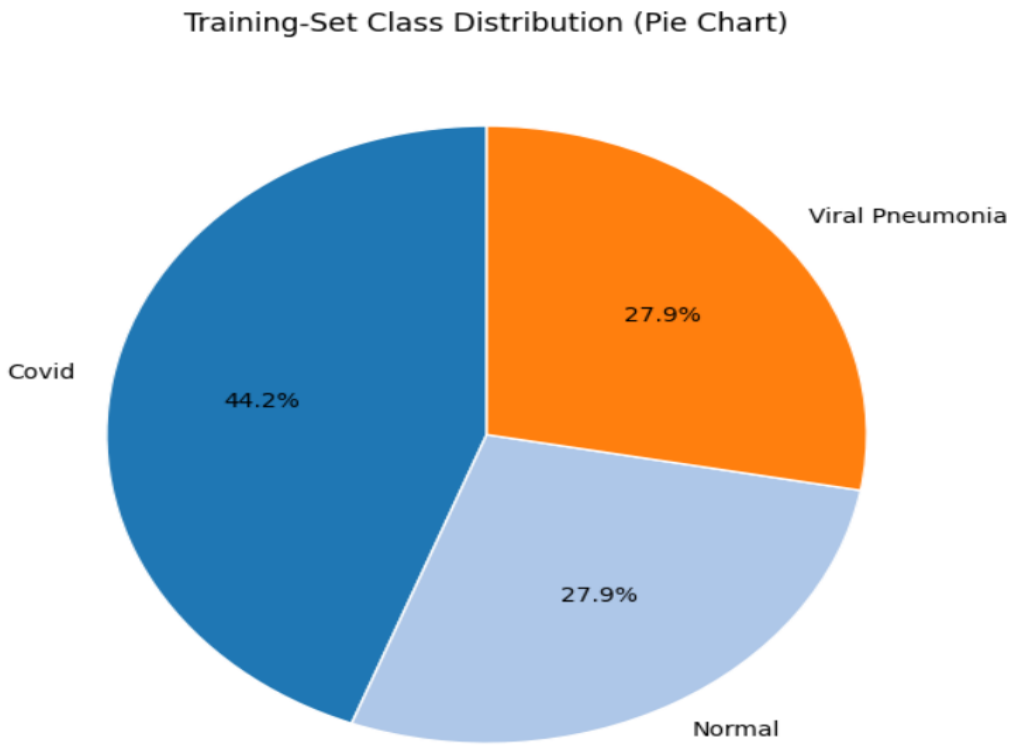

Figure 1 : visualizing the class imbalance

Details of GAN Architectures & Training

1. LSGAN

Least Squares GAN (LSGAN) was utilized to address class imbalance in my dataset by synthesizing minority class images. The model consists of a generator and a discriminator that both utilize fully connected layers. The generator takes a 100-dimensional latent noise vector and maps it through a series of linear and batch-normalized layers with ReLU activations, ultimately outputting a flattened RGB image of size(3,64,64) , then passed through a Tanh activation to scale pixel values from -1 to 1. The discriminator also uses linear layers with Leaky ReLU (rate 0.2), eventually resulting in a single output neuron without the use of sigmoid, since LSGANs use mean squared error (MSE) loss instead of binary cross-entropy. The generator is trained to minimize the MSE between synthetic images and real labels , and the discriminator is trained to minimize the MSE for real (label 1) and synthetic (label 0) images. It was trained on a subset of two minority classes—Normal and Viral Pneumonia—for 10 epochs to generate enough synthetic examples to balance the majority class (Covid). Synthetic images were stored following training and

subsequently combined with the original dataset to achieve a balanced classification pipeline.

2. CGAN

A Conditional GAN (CGAN) was used to enable class-conditioned image generation for data augmentation. The generator and discriminator utilize class labels via embedding layers so that the image type can be controlled. The generator accepts a 100-dimensional noise vector and a class label, which are embedded and concatenated and then fed through dense layers with ReLU and batch normalization, ultimately being reshaped to an image tensor of size(3,64,64). The discriminator receives a flattened image and an embedded label, concatenates them, and passes them through dense layers with Leaky ReLU to give a probability of whether the input image is real and belongs to the given class. Binary cross-entropy loss was used for both the networks and Adam optimizer with learning rate (0.0002). The CGAN was trained on 10 epochs, in which synthetic images were generated occasionally by class to be used for testing. After training, additional images were synthesized for underrepresented classes to balance the dataset so that each of the three classes—Covid, Normal, and Viral Pneumonia—had equal sample sizes for subsequent CNN training.

3. Vanilla GAN

A Vanilla GAN model was utilized to address class imbalance. The generator network, consisting of transposed convolutional layers, mapped a 100-dimensional latent noise vector to realistic RGB images of size (3,64,64), using batch normalization, ReLU activations, and an ultimate Tanh activation to constrain pixel values between -1 and 1. The discriminator, consisting of convolutional layers with Leaky ReLU (slope=0.2) and batch normalization, classified input images as real or fake via a sigmoid-activated output neuron. Both the networks were trained using Binary Cross-Entropy (BCE) loss and Adam optimizer with a learning rate of (0.0002), betas (0.5, 0.999). Both GANs were trained individually for 10 epochs on batches of 32 images of their corresponding class, going back and forth between training the discriminator on real and generated images and training the generator to fool the discriminator. Each 10 epochs, the sample images were dumped for visual check-up, and during training, each generator was utilized to produce 41 additional images per minority class. The produced images were denormalized, stored on disk, and then combined into the original dataset to give a balanced dataset, which makes more balanced and efficient training of a downstream CNN classifier.

Classifier Setup and Evaluation

A Convolutional Neural Network (CNN) model was created to categorize chest X-ray images into three categories: Covid-19, Normal, and Viral Pneumonia. The model (without balancing the dataset)architecture consists of three convolutional blocks, each of which consists of a Conv2D layer followed by MaxPooling2D for extracting spatial features and downsampling. These are succeeded by a flattening layer and two fully connected Dense layers, ending with a softmax output layer to produce class probabilities. The pictures were resized to 224×224 pixels and normalized to [0,1]. Random rotation, zoom, and horizontal flipping data augmentation techniques were also utilized while training for better generalization and to avoid overfitting.

Data preprocessing of the dataset was performed using an intense pipeline that took various forms of image data—raw NumPy arrays, Python lists, and byte streams—transforming between them to ensure consistency across samples. All images were resized and converted to RGB format before being split into training and validation sets. Since there was no separate test set in the dataset, an 80-20 train-validation split was used wherever required. The model was compiled using the Adam optimizer and sparse categorical crossentropy loss, which is suitable for multi-class classification when one-hot encoded labels are not employed.

Training was conducted for 10 epochs with a batch size of 32. The model trained well, with a final validation accuracy of 90.91% and low validation loss of 0.2125. The training accuracy improved consistently, up to 91.28%, indicating that the model learned good features from the augmented data. However, validation accuracy fluctuated a bit in the later epochs that reflected mild overfitting, but it was trivial as regularization was done using data augmentation and pooling layers.

The classification report also confirmed the performance of the model. It achieved perfect precision and recall (1.00) for class Covid-19 (class 0), reflecting all correct predictions with no false positives or false negatives. For class 1 (Normal class), the model was perfectly recalled (1.00) but was slightly less precise (0.77), reflecting a tendency to misclassify other types as Normal. Class 2 (Viral Pneumonia class) had 0.70 recall, with some instances being overlooked, although precision was perfect at 1.00, with all predicted Viral Pneumonia instances being correct. Overall, the macro average F1-score was 0.90, reflecting nicely balanced performance across both classes.

A confusion matrix was also plotted to visualize the model's performance on actual vs. predicted labels. Training and validation accuracy and loss plots per epoch also revealed

that the model converged well, with both measures steadily improving and validating the choice of network structure and training parameters.

Results & Comparisons

After running all the models, these are the key results:

1. Original Imbalanced Dataset
   Had highest validation accuracy (90.91%) but exhibited imbalanced class performance:
   Optimal precision/recall for Covid-19 (Class 0) due to over-representation.
   Poor recall for Viral Pneumonia (Class 2) (70%), indicating missed predictions for minority class.
   Model would likely have concentrated on majority class (Covid-19), leading to biased predictions.
   2. Vanilla GAN-Balanced Dataset
   Validation accuracy dropped marginally to 87.88%, although class balance was improved:
   Enhanced recall for Viral Pneumonia (75% → 86% F1) compared to the initial dataset.
   Enhanced generator loss across training (G Loss: 3.6070 → 5.4264), reflecting diminishing returns in image quality.
   Trade-off: Reduced accuracy slightly but enhanced generalization across classes.
   3. LSGAN-Balanced Dataset
   Poor accuracy (82.35%) and unstable GAN training (e.g., fluctuating discriminator loss: D Loss: 0.0148–0.1250).
   Good Viral Pneumonia recall (95.24%) but poor Normal (Class 1) precision (58.82% recall).
   LSGAN's generated images fail to capture fine-grained realism, thus affecting CNN discriminability.
   4. CGAN-Balanced Dataset
   Validation accuracy: 86.36%, with solid class-specific performance:
   High precision for all classes (1.00 for Covid-19 and Viral Pneumonia).
   Recall for Normal (Class 1) improved to 100%, yet recall for Viral Pneumonia remained low (65%).
   GAN training was stable (D Loss: 0.0402 in last epoch), but generator loss skyrocketed (G Loss: 3.0568), indicating potential mode collapse.
   The table below shows a comparison between the models.

Table 1:comparsion between the GANs

Balancing Method	Validation Accuracy	Covid-19 (Class 0)		Normal (Class 1)			Viral Pneumonia (Class 2)				Macro Avg F1	GAN Training Loss
		Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1		D Loss (Final)
Original Imbalanced Dataset	90.91%	1	1	1	0.77	1	0.87	1	0.7	0.82	0.9	N/A
Vanilla GAN-Balanced	87.88%	1	0.88	0.94	0.71	1	0.83	1	0.75	0.86	0.88	0.0503
LSGAN-Balanced	82.35%	0.8571	0.9231	0.8889	0.9091	0.5882	0.7143	0.7692	0.9524	0.8511	0.8181	0.125
CGAN-Balanced	86.36%	1	0.92	0.96	0.69	1	0.82	1	0.65	0.79	0.85	0.0402

Observations and Conclusions

- Impact of Balancing :
  Balancing improved minority class recall (e.g., Viral Pneumonia in Vanilla GAN) but at the expense of accuracy.
  The original imbalanced dataset was accurate but biased towards the majority class.
- GAN Quality vs. Performance:
  Vanilla GAN produced the highest-quality balanced results but suffered from generator instability (G Loss: 5.43).
  LSGAN was plagued by convergence issues (D Loss: 0.1250), producing low-quality synthetic images.
  CGAN enabled class-specific generation but showed class-wise performance fluctuations (e.g., low Viral Pneumonia recall).
- Clinical Relevance :
  In high-stakes settings (e.g., Covid-19 diagnosis), the base dataset model performed nearly perfectly but could miss cases of Viral Pneumonia.
  Balanced datasets (e.g., Vanilla GAN) allow safer generalization but at the cost of slight accuracy reductions.

In summary, as balance improved batch equity among classes, none surpassed the original unbalanced test set with respect to simple accuracy. But th ey assisted in rectifying core minorities class prediction skew that is critical to clinical reliability. Stabilizing GAN training (e.g., Wasserstein loss, attention mechanisms) and enhancing synthetic image realism further are aspects future research ought to focus on in order to close the performance gap.