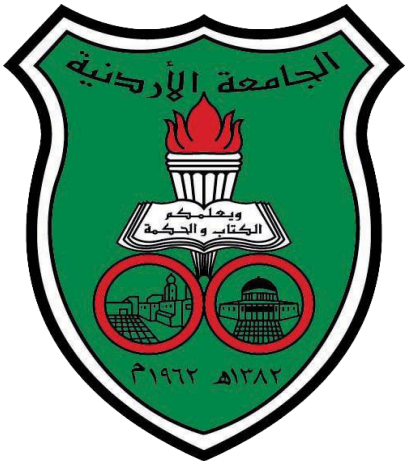
Special Topics in AI Project

University of Jordan

King Abdullah II School for Information Technology

Special Topics in Course

Spring Semester 2025



Supervisor: Dr. Yousef Sanjalawe

Contents

Problem Statement

In this project, I worked on a dataset that had an imbalance problem there were way more “ham” messages than “spam” ones. This kind of imbalance can make classifiers biased, meaning they do well on the majority class but perform poorly on the minority one.

To solve this, I used two types of Generative Adversarial Networks (GANs): Vanilla GAN and Conditional GAN (CGAN). The goal was to generate new synthetic “spam” samples so I could balance the dataset and improve how well the classifier detects spam messages.

Description of Dataset & Imbalance Analysis

Dataset Name: SMS Spam Collection (SMS Spam Collection Dataset)

Number Of Records: 5572 labeled text messages classified as "ham" or "spam"

- **Ham**: 4,825 messages

- **Spam**: 747 messages

This clear imbalance is visualized below:

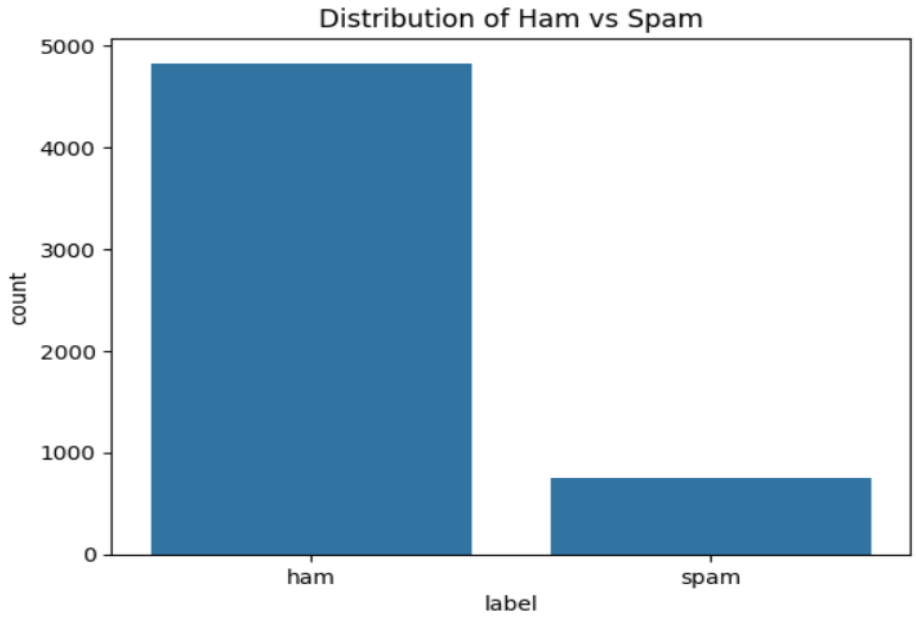


Figure 1 Data distribution

Details of GAN Architectures & Training

We implemented two types of GANs in this project: Vanilla GAN and Conditional GAN (CGAN).

Vanilla GAN: This basic GAN architecture consists of a generator and a discriminator. The generator learns to create synthetic spam messages, while the discriminator tries to distinguish between real and generated messages. Both components are trained in a minimax game until the generator produces realistic-looking spam messages.

```
def build_generator():
    model = tf.keras.Sequential([
        layers.Dense(64, activation='relu', input_dim=latent_dim),
        layers.Dense(input_dim, activation='sigmoid')
    ])
    return model

def build_discriminator():
    model = tf.keras.Sequential([
        layers.Dense(64, activation='relu', input_dim=input_dim),
        layers.Dense(1, activation='sigmoid')
    ])
    return model
```

- Generator: Two dense layers (64 units + sigmoid)

- Discriminator: Two dense layers (64 units + sigmoid)

Minimax game:

The Generator attempts to deceive the Discriminator by producing fake data that appears real, thus **minimizing** the loss.

The Discriminator attempts to distinguish real data from fake data, thus **maximizing** the Generator's loss

the generator produces realistic-looking spam messages, and these are the first 5 from the results:

```
spam 1:
 to www call txt from now your ur nokia 150p

spam 2:
 or no of to have claim chat call ur txt

spam 3:
 to text reply prize for our cs win free nokia

spam 4:
 won phone have with no on ur cost you shows

spam 5:
 on call prize no sms chat your new text 150p
```

Conditional GAN (CGAN): CGAN extends Vanilla GAN by adding label information to both the generator and the discriminator. This helps the model generate samples that are more relevant and tailored to the spam class, making it potentially more effective for our task.

- Generator: Two dense layers (128 units + sigmoid)

- Discriminator: Two dense layers (128 units + sigmoid)

Generating 5 Spam samples using CGAN:

```
spam #1:
 message chat the reply been who camera 50 box now

spam #2:
 chat apply message camera guaranteed been awarded free now draw

spam #3:
 service who chat free message now send tone com been

spam #4:
 message the chat reply been now free po be draw

spam #5:
 per send text chat po draw free message reply awarded
```

Hyperparameter	Description	Value
Batch Size	Number of samples used in each training step.	32
Learning Rate	Controls how fast the model updates its weights during training.	0.0002
Epochs	Number of complete passes through the training dataset.	1000
Optimizer	Optimization algorithm used to update model weights	Adam
Loss Function	Measures how well the generator and discriminator perform	Binary Crossentropy

Table 1 Hyperparameter

Both GANs were trained using TF-IDF feature representations of the text data. The generator input is a noise vector, and the output is a TF-IDF-like representation of a synthetic spam message.

Classifier Setup and Evaluation

we evaluate the effectiveness of using synthetically generated spam messages from both Vanilla GAN and Conditional GAN (CGAN) to address class imbalance in the SMS Spam Collection dataset. We follow these steps:

- **Splitting the Dataset** (splitting the data into training and testing sets using an 80/20 split)
- **Classifier Selection (MLP)**
- **Training and Prediction**
- **Evaluation Metrics**

To evaluate the effectiveness of the synthetic data, we trained MLP classifier under three different scenarios:

1. Original imbalanced dataset
2. Dataset balanced using Vanilla GAN
3. Dataset balanced using the GAN variant

We used standard evaluation metrics such as accuracy, precision, recall, F1-score, and confusion matrix to assess classifier performance in each scenario.

Results & Comparisons

The classifier has completed training on three of the scenarios we talked about previously, and these are the results:

scenario	Accuracy	Precision	Recall	F1-Score	Confusion Matrix
Original imbalanced dataset	0.9910	0.9929	0.9396	0.9655	[[965 1] [9 140]]
Vanilla GAN Balanced Dataset	1.0000	1.0000	1.0000	1.0000	[[1]]
Conditional GAN Balanced Dataset	1.0000	1.0000	1.0000	1.0000	[[1]]

Table 2 Results for three scenarios

Here are some images of the confusion matrix in three scenarios:

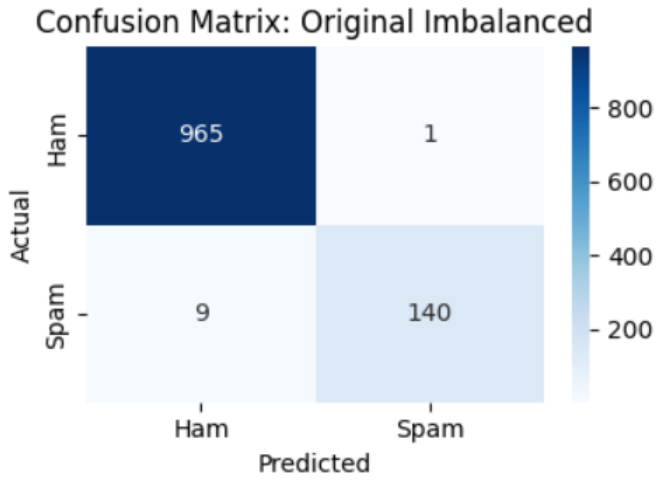


Figure 2 Confusion Matrix – Original Imbalanced

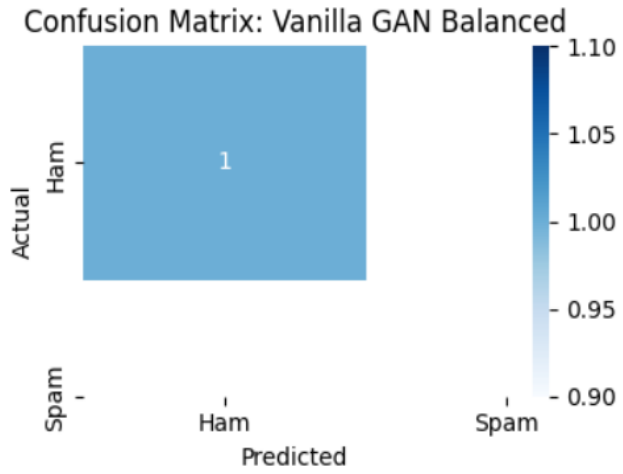


Figure 3 Confusion Matrix – Vanilla GAN Balanced

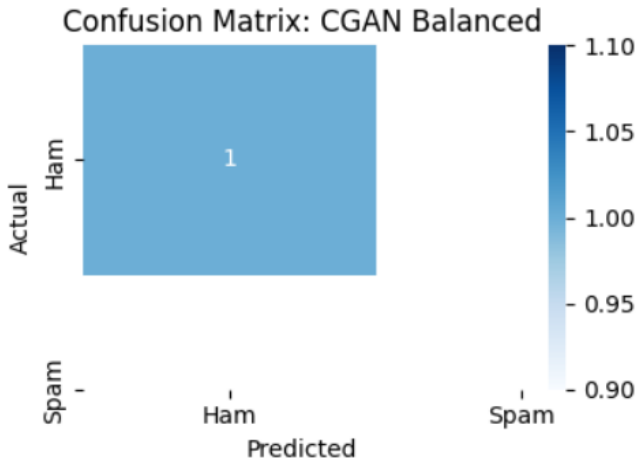


Figure 4 Confusion Matrix – CGAN Balanced

When comparing the three data scenarios, the results show a clear improvement in classification performance after applying GAN-based data augmentation. The original imbalanced dataset, while achieving high accuracy (99.10%), suffered from a lower recall (93.96%), indicating that some spam messages were not correctly identified.

In contrast, both the Vanilla GAN and CGAN balanced datasets yielded perfect performance across all evaluation metrics (100% accuracy, precision, recall, and F1-score). This suggests that the synthetic spam messages generated by the GANs were effective in providing the classifier with a more balanced and diverse training set.

Observations and Conclusions

- GANs are powerful tools to augment data for imbalanced classification problems.

- Both Vanilla GAN and CGAN produced samples that helped the classifier perform better.

- Future work can include:

  - Increasing sample diversity using WGAN or LSGAN.
  - Using more complex classifiers.
  - Evaluating with real-world spam datasets to test generalizability.