# Text-to-Image GAN Project

This repository contains the code and a selection of results from a project focused on training a Generative Adversarial Network (GAN) for text-to-image generation using the COCO dataset. Due to storage limitations, this repository includes a curated set of outputs that demonstrate the key aspects of each training phase, rather than exhaustive data.

## Links to Complete output Files and Codes

* overfit\_notebook : [kaggle overfit link](https://www.kaggle.com/code/arthasmenethiltest/imagegenfromtext-overfit)
* optuna\_notebook [kaggle optuna link](https://www.kaggle.com/code/machinelearningmain/imagegenfromtext-optuna-exploration)
* final\_training\_notebook [kaggle final training link](https://www.kaggle.com/code/arthasmmenethil/imagegenfromtext-final-training)

## Repository Structure

The repository is organized into three main folders:

* final\_training\_notebook Contains the code and results from the final training run of the GAN model.
* optuna\_notebook Demonstrates the hyperparameter optimization process using Optuna.
* overfit\_notebook Shows an overfitting test to verify the model's ability to learn from a small dataset.

## Contents and Space-Saving Considerations

Due to the large size of model checkpoints (`.pth` files), extensive logs, and numerous intermediate results, the following decisions were made to keep the repository size manageable:

### `final\_training\_notebook`

* Included:
* Generated Images: Sample images generated by the final trained GAN model, showcasing the model's ability to create images from text descriptions.
* Losses: Plots of the generator and discriminator losses during training, illustrating the training dynamics and convergence.
* Excluded:
* Model Checkpoints (`.pth`): The trained model files were too large to include in this repository.
* Full Training Logs: Detailed training logs, which can be extensive, were omitted to conserve space.
* MLflow Runs: Full MLflow run data, including metrics, parameters, and artifacts, are not included.

### `optuna\_notebook`

* Included:
* Model Architecture: Description or code snippets defining the final model architecture selected through hyperparameter tuning.
* Optuna Losses: Plots or summaries of the loss curves obtained during the Optuna optimization process, demonstrating how hyperparameters were evaluated.
* best\_params.txt A text file listing the best hyperparameters found by Optuna.
* Excluded:
* Intermediate Model Checkpoints: Model checkpoints from each Optuna trial are not included.
* All Hyperparameter Configurations: Full details of every hyperparameter combination tried by Optuna are omitted. Only the best set is included.

### `overfit\_notebook`

* Included:
* Exploratory Data Analysis (EDA) Visualizations: Selected visualizations from the initial exploratory analysis of the COCO dataset, providing insights into the data.
* Overfit Real vs. Fake Images: Visual comparisons of real and generated images from the overfitting test, highlighting the model's ability to memorize and reproduce training samples.
* Losses (MSE, Reconstruction Error): Plots of Mean Squared Error (MSE) and other reconstruction error metrics during the overfitting test, illustrating the model's learning on a small dataset.
* Excluded:
* Full COCO Dataset Visualizations: Due to space limitations, only a representative subset of EDA visualizations is provided.
* Overfit Model Checkpoints: The `.pth` files from the overfitting test are not included.
* Fold Data: Data related to cross-validation folds, if used during development, is not included.

## General Exclusions

* Model Checkpoints (`.pth` files): All model checkpoints were omitted due to their large file sizes.
* Extensive Logs: Detailed log files from training, validation, and hyperparameter tuning are not included.
* MLflow Runs: Complete MLflow tracking data is not included in this repository.

## Getting Started

To reproduce the results, you will need to:

1. 1. Set up the environment using the provided `requirements.txt` (if applicable).
2. 2. Obtain the COCO dataset and preprocess it as described in the notebooks.

3. Run the notebooks in the following order:

1. 1. `overfit\_notebook` (to understand the overfitting behavior)
2. 2. `optuna\_notebook` (to perform hyperparameter optimization)

3. `final\_training\_notebook` (to train the final model,based on the best hyperparameters extracted by the optuna notebook)

## Notes

* This repository is intended to provide a concise overview of the project and its key results.
* The code can be used as a starting point for further experimentation and development.
* Refer to the individual notebooks for more detailed explanations of the code and results.
* For access to the full outputs and code, please visit the associated Kaggle notebooks linked above.

## Contact

Vasilis Tsavalias

vas.tsavv@gmail.com