**Test Plan Document for ArtiGaN**

SP22 CSCI-P446 Capstone RF 6

Nathanael L. Mann, Mark Bridgewater, Joshua Carrier

Sponsor: Chris Sexton

25 January 2022

**1 Introduction**

This is a testing document for the Arti-GaN project. In this document, we will cover the intended functionality of Arti-GaN, and propose the methods by which the functionality of the project will be ensured.

**2 Test Objectives**

The purpose of the testing is to ensure that the GAN is learning and doing so at a reasonable rate.

**3 Scope**

**3.1 Inclusions**

We will be testing the mathematical results of the GaN to ensure that the learning variables are trending in the correct direction.

**3.2 Exclusions**

We will not be judging the images based on their quality, due to a vast breadth of unpredictability in what kinds of images will be produced.

**5 Test types Identified**

1. Test the confidence of the Generator Agent
2. Test the confidence of the Discriminator Agent

**6 Problems Perceived**

There is concern that the AI will not generate images that “appear” reasonably similar to the test set within a reasonable amount of time, given the hardware available for testing. Additionally, the originally proposed webpage as a front-end user-interactive platform, as of submitting this document, has not come to fruition, and a number of “missing” tests will be attributed to the lack of this UI.

**7 Architecture**

The generator currently tests the input size of the image at every layer in the GAN. This assures that the images that are produced are of the correct dimensions. To train the GAN, we made our own loss functions. When training, we run a batch of real images through the discriminator and save the output. We then generate a batch of the same size of fake images and do the same for those images. To calculate the loss for the discriminator, we compare the output of real images to an array of 1’s and save the difference. This represents the loss of the real images. Similarly, we compare the output for the fake images to an array of 0’s and save the difference. The sum of these two numbers gives us the total loss of the discriminator. To calculate the loss of the generator, we calculate the difference of the fake output to an array of 1’s. We use the discriminator loss and the generator loss to calculate the values for our gradient values to adjust the GAN correctly.

**8 Environment**

Arti-GaN was tested and trained in the following environment:

* OS Version - MINGW64\_NT-10.0-22538
* Free Memory - 10967700 kB
* CPU - Intel(R) Core(TM) i7-7500U CPU @ 2.70GHz

**9 Assumptions**

In order for accurate results, the system needs to be run for an extended period of time. Currently, we are in the process of determining the range of time necessary for optimal results. Using non-tested datasets may result in unexpected output images. The tested data sets are as follows: Bob Ross Paintings <https://github.com/jwilber/Bob_Ross_Paintings/tree/master/data/paintings>

**10 Functionality**

The main function of Arti-GaN is to produce images similar to those in the test set.

**10.1 Constraints and Resolutions**

| **Parameter** | **Constraints** | **Limitations** |
| --- | --- | --- |
| Time to run | We have limited hardware to run tests | We have limited time to run tests. |

**10.2 Risk Identified & Mitigation Planned**

The program may run for an extended period of time, and not produce meaningful results. This cannot easily be accounted for until the program has run for a long enough period of time, potentially resulting in wasted time testing. There is little that can be done about this given the hardware constraints, and time limitations of the project.

**10.3 Test Strategy**

We intend to run various iterations of the program over the course of the testing period, making changes where necessary to encourage the results desired. Tests will run in ranges from a few hours to a full day.

**10.4 Automation Plans**

Starting the program takes minimal effort and only needs to be done at a maximum of once per day, so no automation is necessary.

**10.5 Deliverables**

The program will output images at the end of each testing cycle, those images should show a trend of progress as the test goes forward, and refinements are made to the algorithms.

**11 Security**

The Security section has been neglected since the only portion of the project that has been completed is the Open-Source back-end of the program. The front-end website would have comprised the entirety of the tests in this section.

**12 Performance**

The performance of the machine is essentially measured in the time it takes to produce an image (covered in the functionality section) and the trend of the learning curve.

**12.1 Constraints and Resolutions**

| **Parameter** | **Constraints** | **Limitations** |
| --- | --- | --- |
| Discriminator Confidence | Random Elements | Time |
| Generator Confidence | Random Elements | Time |

**12.2 Risk Identified & Mitigation Planned**

A common issue among Artificial Intelligence programs is that of “two idiots talking to each other” which results in program outputs that appear as if no learning has occurred. In order to best mitigate this, the progress of the algorithm is checked at regular intervals during testing, and if a lack of improvement, or backward progress is identified, the program can be terminated, tweaked and restarted again.

**12.3 Test Strategy**

These values are manually checked at the end of the testing period, as the processing power needed to run the rest of the program would be halted greatly if automatic checking was implemented. While not the most efficient, it is clear when no progress is being made by the algorithm, and tweaks have to be made manually regardless, so having an individual check the progress is actually more efficient than stopping the program when a lack of progress is detected.

**12.4 Automation Plans**

We have no plans to automate this test, beyond calculating and outputting the results so they can be checked by an observer.

**12.5 Deliverables**

The data outputted here could be delivered, but would lack significant meaning to an untrained observer.

**13 Usability**

This category also falls under the lack of Website clause. The back-end program’s usability can be defined on the basis of the other categories, so tests in those sections cover back-end usability.

**14 Compatibility**

We have no current plans to test compatibility for the backend. Front-end compatibility would involve making sure the webpage is accessible on various browsers, and formats (such as mobile or desktop), but due to a lack of a webpage, these things need not be tested.

**15 Test Team Organization & Schedule**

Each member of the team is responsible for testing the project application that they constructed. Mann and Carrier work on the GaN, while Bridgewater is in charge of the Website.

Out team performs tests through each week of the project's duration, and reports with each other on Mondays and Fridays, with the majority of test results being reported on Fridays, while Mondays are reserved for more conversational meetings to discuss the project.

**16 Defects Classification Mechanism**

| **Types of Defects** | **Functionality** | **Performance** | **Security** | **Usability** | **Compatibility** |
| --- | --- | --- | --- | --- | --- |
| **Critical** | 1 Week of testing, shows no results | Progress is not made. | N/A | N/A | N/A |
| **Major** | 1 Day of testing shows no results | Significant Progress is not made. | N/A | N/A | N/A |
| **Minor** | Several Hours of testing show no results | Images contain imperfections. | N/A | N/A | N/A |
| **Cosmetics** | N/A | Images produced are “incorrect” | N/A | N/A | N/A |

**16.1 Defects Logging and Status Changing Mechanism**

Logs are kept in weekly meetings on Friday to track the progress of the testing.

**16.2 Turn Around Time for defect fixes**

Depending on the complexity of the defect, the turnaround time could vary. Anything taking longer than 1 day (of work) to fix is considered abnormal.

**17 Release Criteria**

If the project passes testing, and produces images of reasonable quality, the project is suitable for release. Release, at the time of writing this test documentation, involves essentially releasing an open-source program that other individuals or groups can use in their own research and/or projects.