# Overall Test Plan

The goal of the testing which will be performed in the project is to verify that the functional building blocks of the system work as expected to demonstrate the authenticity of the project’s research findings. The first piece of this will be to test synthetic data generation components to ensure that synthetic data conforms is of the correct format and that user-specified distributions of features are present in the generated data. Also key to the research is the computation of different dataset distance metrics (e.g., Bhattacharyya distance), each of which will need unit-tested to ensure their correctness. The pipeline for ingesting data (both measured and synthetic), constructing and training a computer-vision model, and compiling results will require unit-testing at each stage and integration testing to ensure each piece is communicating properly with the others, as well. Finally, we intend to primarily focus on normal cases due to the great control we will have over the operating inputs and environment during the computation phase of the research.

# Test Case Descriptions

OCG1.1 Operating Characteristic Generator 1

OCG1.2 Ensure system correctly ingests and parses OC (Operating Characteristic) requirements

OCG1.3 OC requirements are specified in a excel schema defining the range of accepted values, type, and nature of each OC. Correct ingestion of this schema into a PGM (Probabilistic Graphical Model) will be verified via software introspection to ensure that the system is capable of generating valid sets of OCs for the Synthetic Data Generator to later use.

OCG1.4 An excel-formatted OC schema

OGC1.5 An in-memory PGM

OGC1.6 Normal

OGC1.7 Whitebox

OGC1.8 Functional

OGC1.9 Unit

OGC2.1 Operating Characteristic Generator 2

OGC2.2 Ensure correct validation of OC schemas

OGC2.3 OC requirements are specified in excel documents, but as these as user-editable, their conformance to their corresponding schema specification needs to be verified. A method will accept an excel file handle and yield a Boolean specifying if it is a valid spec. We will feed in a variety of schemas to ensure the module yields the manually verified result.

OGC2.4 An excel-formatted OC schema

OGC2.5 A Boolean result

OGC2.6 Abnormal

OGC2.7 Blackbox

OGC2.8 Functional

OGC2.9 Unit

OGC3.1 Operating Characteristic Generator 3

OGC3.2 Ensure valid generation of OCs from PGM

OGC3.3 From a validated PGM, test the sampling engine to ensure valid “bags” of operating characteristics are produced. This will involve generating a large (n > 500) samples, then iterating these to ensure none of them have parameters out of range.

OGC3.4 A validated PGM

OGC3.5 A Boolean Result

OGC3.6 Normal

OGC3.7 Blackbox

OGC3.8 Functional

OGC3.9 Unit

OGC4.1 Operating Characteristic Generator 4

OGC4.2 Ensure correct storage of OC-samples to persisted database

OGC4.3 For a given run of the OC Generator, verify that the persistence mechanisms for saving the generated OCs and loading them into memory again are functional. This will involve generating a large (n > 500) samples, saving them out to a database, then loading the samples from the database and comparing the two sets of samples before and after persistence.

OGC4.4 A large number (n > 500) of samples, in-memory

OGC4.5 A Boolean result

OGC4.6 Normal

OGC4.7 Blackbox

OGC4.8 Functional

OGC4.9 Integration

OGC5.1 Operating Characteristic Generator 5

OGC5.2 Ensure correct communication between different sub-components of the OC Generator

OGC5.3 Begin by reading in a human-verified OC schema, construct a PGM from it, sample a large (n > 500) number of times, then persist those samples to a database. Presence of the correct number of additional samples in the database, together with verifying the reasonability of a random, small number of new samples in the database will suffice to ensure working operation.

OGC5.4 A human validated OC schema (excel file)

OGC5.5 A Boolean value; *n* samples in a persisted database

OGC5.6 Normal

OGC5.7 Blackbox

OGC5.8 Functional

OGC5.9 Integration

OGC6.1 Operating Characteristic Generator 6 (Performance)

OGC6.2 Ensure adequate speed of OC sample generation

OGC6.3 The OC Generator will be timed to determine how much compute resources are expended to create a large (N > 10000) number of samples and persist them to a database. A goal of at least 30 samples per second, amortized, is considered passing.

OGC6.4 Path to an OC schema excel file

OGC6.5 A populated database of samples

OGC6.6 Normal

OGC6.7 Blackbox

OGC6.8 Performance

OGC6.9 Integration

SDG1.1 Synthetic Data Generator 1

SDG1.2 Ensure ingestion of OC sample data into memory

SDG1.3 The synthetic data generator will read in a sample from the persisted database, and software introspection will be used to ensure it’s in-memory representation is correct

SDG1.4 Path to persisted database

SDG1.5 N/A

SDG1.6 Normal

SDG1.7 Whitebox

SDG1.8 Functional

SDG1.9 Unit

SDG2.1 Synthetic Data Generator 2

SDG2.2 Ensure correctness of generated synthetic data

SDG2.3 Feed synthetic data generator module a valid OC sample and observe production and writing of a valid .NITF file which respects the provided sample.

SDG2.4 An operating characteristic sample

SDG2.5 A valid .NITF file, persisted to the filesystem

SDG2.6 Normal

SDG2.7 Whitebox

SDG2.8 Functional

SDG2.9 Unit

DL1.1 Dataloader 1

DL1.2 Ensure correct conversion of .NITF SAR data into single-channel raster data, suitable for ingestion into deep neural networks

DL1.3 Provide the dataloader a directory of .NITF Sar data, convert each into a single-channel raster format, and write the data out into a JPEG format on the disc. We will manually examine a random sampling of the written JPEGs to ensure their reasonability.

DL1.4 Path to directory of .NITF SAR files

DL1.5 A directory of JPEG images corresponding to each .NITF

DL1.6 Normal

DL1.7 Blackbox

DL1.8 Functional

DL1.9 Unit

DM1.1 Distance Metric 1 – Bhattacharyya

DM1.2 Ensure correct computation of Bhattacharyya Distance for single dimensional, discrete distributions

DM1.3 Using two fixed distributions over the same domain with known Bhattacharyya distance, compute it and compare the results of our module with the known value.

DM1.4 Two sets of numeric, single-dimensional samples over the same domain

DM1.5 The Bhattacharyya distance between the two samples

DM1.6 Normal

DM1.7 Blackbox

DM1.8 Functional

DM1.9 Unit

DM2.1 Distance Metric 2 – KL

DM2.2 Ensure correct computation of Kullback-Leibler Divergence for single dimensional, discrete distributions

DM2.3 Using two fixed distributions over the same domain with known KL Divergence, compute it and compare the results of our module with the known value.

DM2.4 Two sets of numeric, single-dimensional samples over the same domain

DM2.5 The KL Divergence between the two samples

DM2.6 Normal

DM2.7 Blackbox

DM2.8 Functional

DM2.9 Unit

DNN1.1 Deep Neural Network 1 – Trainer

DNN1.2 Ensure we can train computer vision recognition model on a raster dataset

DNN1.3 Using a dataset of rasterized SAR imagery, train a DNN and determine if its performance on a reserved validation set is reasonable given the network’s size and complexity.

DNN1.4 A dataset, comprised of a folder of JPEG-encoded SAR imagery

DNN1.5 Validation scores of the network, when it performs on a validation dataset

DNN1.6 Normal

DNN1.7 Blackbox

DNN1.8 Functional

DNN1.9 Unit

# Test Case Matrix

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Normal/Ab. | Black/Whitebox | Funct/Perform | Unit/Integr. |
| OGC1 | Normal | Whitebox | Functional | Unit |
| OGC2 | Abnormal | Blackbox | Functional | Unit |
| OGC3 | Normal | Blackbox | Functional | Unit |
| OGC4 | Normal | Blackbox | Functional | Integration |
| OGC5 | Normal | Blackbox | Functional | Integration |
| OGC6 | Normal | Blackbox | Performance | Integration |
| SDG1 | Normal | Whitebox | Functional | Unit |
| SDG2 | Normal | Whitebox | Functional | Unit |
| DL1 | Normal | Blackbox | Functional | Unit |
| DM1 | Normal | Blackbox | Functional | Unit |
| DM2 | Normal | Blackbox | Functional | Unit |
| DNN1 | Normal | Blackbox | Functional | Unit |