

Assignment Cover Sheet



- The information on this coversheet will be included in Turnitin's similarity analysis; however, your lecturers are aware of this and will disregard it.

Student Details									
Student Number									
Family Name							Given Name		

Unit Details					
Unit Code	CSG3303	Unit Title	Applied IT Project		
Name of Lecturer				Due Date	
Topic of Assignment	Project Report	Group or Tutorial (if applicable)		Nil – Individual Project	
Course				Campus	
<p>I certify that the attached assignment is my own work and that any material drawn from other sources has been acknowledged. This work has not previously been submitted for assessment in any other unit or course. Copyright in assignments remains my property. I grant permission to the University to make copies of assignments for assessment, review and/or record keeping purposes. I note that the University reserves the right to check my assignment for plagiarism. Should the reproduction of all or part of an assignment be required by the University for any purpose other than those mentioned above, appropriate authorisation will be sought from me on the relevant form.</p>					

Manual Submission				
If handing in an assignment in a paper or other physical form, sign here to indicate that you have read this form, filled it in completely and that you certify as above.				
Signature			Date	1/11/2019
Electronic Submission				Office Use Only
<p>OR, if submitting this paper electronically as per instructions for the unit, place an 'X' in the box below to indicate that you have read this form and filled it in completely and that you certify as above. Please include this page with your submission. Any responses to this submission will be sent to your ECU email address.</p>				
Agreement	<input checked="" type="checkbox"/> select check box		Date	
<p>For procedures and penalties on late assignments please refer to the University Admission, Enrolment and Academic Progress Rules - rule 24, and the ECU Course and Unit Delivery and Assessment Policy</p>				

The information on this coversheet will be included in Turnitin's similarity analysis; however, your lecturers are aware of this and will disregard it.

This section is for lecturers to complete. It is intended to give you feedback on your written communication skills and, unless otherwise stated, does not earn marks towards your grade.

The ECU English Language Proficiency Measure (Feb 2014)				
Levels of proficiency	Low proficiency	Developing proficiency	Moderate proficiency	High proficiency
Aspects of writing (Indicate with an X main area(s) needing improvement)	Incorrect or inappropriate aspects of writing obscure meaning in many places. Significant editing needed to clarify the meaning along with extensive proofreading to correct technical errors.	Incorrect or inappropriate aspects of writing obscure meaning in some places. Some editing needed to clarify the meaning, along with extensive proofreading to correct technical errors.	Aspects of writing are mostly accurate. Mistakes rarely affect clarity of meaning. Minor editing needed to clarify the meaning, along with careful proofreading to correct technical errors.	Aspects of writing are appropriate and optimally constructed, allowing clarity of meaning. Meaning is clear and needs only a light proofread to correct technical errors.
Sentence structure <input type="checkbox"/> 1. sentence completeness <input type="checkbox"/> 2. sentence length <input type="checkbox"/> 3. phrase/clause order <input type="checkbox"/> 4. use of conjunctions <input type="checkbox"/> 5. word order <input type="checkbox"/> 6. punctuation				
Word use <input type="checkbox"/> 7. word choice <input type="checkbox"/> 8. word form <input type="checkbox"/> 9. word omission/redundancy <input type="checkbox"/> 10. verb tense/agreement <input type="checkbox"/> 11. spelling <input type="checkbox"/> 12. apostrophes				
Sentence Structure 1. Sentence completeness: sentence includes subject, verb a complete thought. 2. Sentence length: length is appropriate to context or discipline. 3. Phrase/clause order: parts of the sentence (phrases and clauses) are ordered logically. 4. Use of conjunctions: linking words are used correctly to show the relationship between ideas. 5. Word order: words are ordered correctly in a sentence. 6. Punctuation: the correct use of full stops, commas, semicolons, colons and capitals.		Word Use 7. Word choice: words are correct and appropriate for the context. 8. Word form: correct part of speech is used, e.g., [to] affect / [the] effect. 9. Word omission/redundancy: words should not be missing or be unnecessarily repetitive. 10. Verb tense/agreement: correct use of verbs that indicate time and correct word forms that agree grammatically with other words in the sentence. 11. Spelling: correct spelling is used. 12. Apostrophes: indicate ownership or contraction.		

The information on this coversheet will be included in Turnitin's similarity analysis; however, your lecturers are aware of this and will disregard it.

CSG3303 – Applied IT Project

Optometer

Analytical Project Report

Semester 2, 2019

Due at 9:00am (UTC +8) on 1 November 2019

github.com/dancingborg/ECU_CSG3303_Optometer

@dancingborg

github.com/dancingborg/ECU_CSG3303_Optometer

1.0 Table of Contents

1.0	Table of Contents	1
2.0	Abstract	3
3.0	Introduction	4
4.0	Background	4
4.1	Problem Definition	4
4.2	Problem Significance & Complexity	5
4.3	Existing Work & Approaches	5
4.4	Results of Other Technologies	6
5.0	Optometer	6
5.1	Scope	6
5.2	Technology Choice	7
5.2.1	GAN	7
5.2.2	SRGAN	7
5.2.3	ESRGAN	7
5.3	Dataset Generation	8
5.3.1	Image Template	8
5.3.2	Generating a Dataset	8
5.3.3	Typeface Selection	9
5.3.4	Image Synthesis	9
5.3.5	Improvements to Synthesis	10
5.3.6	Dataset Preparation	11
5.4	Adaptation of BasicSR	11
6.0	Method	13
6.1	Model Comparison	13
6.2	Increasing Iterations	13
6.3	Metrics	13
6.3.1	PSNR	14
6.3.2	SSIM	14
6.3.3	Improvement Score	15
7.0	Results	15
7.1.1	Iterations Over 1000 Images	16
7.1.2	Iterations Over 45178 Images	17
7.1.3	Learning Rate for 1000 Images and 1000 Iterations	18
7.1.4	Iterations for 1000 Images at 0.0004 Learning Rate	19

7.1.5	Effect of Input Image Degradation.....	20
8.0	Discussion.....	21
8.1	Effect of Reduced Input Image Quality	21
8.2	Effect of Learning Rate.....	22
8.3	Effect of Increasing Iterations	23
8.4	Shortfall of Synthetic Data	25
8.5	Effects of Lossy Input	26
8.6	Comparison to Other Published Work.....	27
9.0	Conclusion.....	28
10.0	References	29
11.0	Appendix 1: Sample Image of Iteration Tests over 1000 Training Images	1
12.0	Appendix 2: Sample Image of Iteration Tests over 45178 Training Images	3
13.0	Appendix 3: Sample Images of Learning Rate Tests of 1000 Images over 1000 Iterations	5
14.0	Appendix 4: Sample Images of Iteration Tests for 1000 Images and 0.0004 Learning Rate	7
15.0	Appendix 5: Quality Comparison as Input Image Degrades	9

2.0 Abstract

The objective of the Optometer project was to test the hypothesis that when executing on an image of a real license plate, a SRGAN trained on synthetic license plates as opposed to generic images will produce a better-quality output. The project was for CSG3303 – Applied IT Project at Edith Cowan University, as the culmination of all preceding learning. The main topics of the project are Super Resolution Generative Adversarial Networks (SRGANs), and the image quality loss metrics of Peak Signal to Noise Ratio (PSNR) and Structural Similarity (SSIM). To test the hypothesis, the Optometer project generated 50000 standard Western Australian license plates, and used a deduplicated subset for training, validation, and evaluation of pre-trained SRGAN models. The project limited license plates to the specific subset so that the datasets came within time and computing resource limitations. To evaluate the hypothesis, the project changed variables such as iteration count, and measured performance by average PSNR and SSIM over a control group of 100 license plates. Experiments showed that although there were some local maxima and minima within the variable ranges, the result trends strongly indicated the usefulness of conducting further experiments at the higher iteration counts that were beyond the resources of this project. At the default training rate, the dataset size did not have a significant effect on model performance across the range of iterations tested. The SRGAN trained with synthetic data highly specific to the problem domain yielded subjectively and objectively better-quality outputs than the SRGAN trained on generic images. This tended to prove the hypothesis of the project.

A copy of all source code, adaptations and original work for this project is located at the following link:

github.com/dancingborg/ECU_CSG3303_Optometer

3.0 Introduction

This report is the primary deliverable for the teaching unit CSG3303 – Applied IT Project, at Edith Cowan University. The purpose of this report is to explain the research conducted by the Optometer project (this project). An Optometer is an ophthalmic device for measuring eyeglass prescriptions. This project has the name Optometer because the goals of the project are analogous to the function of an Optometer.

The hypothesis of this project is that when executing on an image of a real license plate, a SRGAN trained on synthetic license plates as opposed to generic images will produce a better-quality output. To test this hypothesis, this project generated a synthetic license plate dataset, and tested pre-trained models on a control group of license plates to develop a baseline PSNR and SSIM. To improve on the baseline, this project tuned a SRGAN using the synthetic license plates dataset. To evaluate this hypothesis, this project executed each trained model on a control group of license plates and calculated the PSNR and SSIM to measure the effectiveness of the training and tuning.

Where possible, this project used and adapted existing tools for training, running, and evaluating the performance of the SRGANs. Optometer constrained the problem domain to a small subset of vehicle license plates and used only synthetic data. This report begins with a background of the problem, its significance, and its complexity, before reviewing existing research, alternative approaches, and the results of these alternatives. This report introduces the Optometer project with an overview of scope and technology choices, before showing the method used to set up the system.

4.0 Background

Closed Circuit Tele-Vision (CCTV) systems are prevalent worldwide for the purpose of conducting video surveillance of places, vehicles, people, and things. Identifying subjects is a common use of CCTV systems, and the identification of vehicles by way of their vehicle license plates is a useful activity for government officials and private citizens (Sarfraz et al., 2013). Automatic recognition of license plates by way of reading the plate's registration number is the subject of diverse contemporary research under the topic of Automatic License/Number plate Recognition (ALPR/ANPR) (Lotufo, Morgan, & Johnson, 1990). Machine Learning (ML) has provided much benefit to this field of research, in detecting license plates in still or moving images and reading the license plate by recognising the characters depicted through inference (Sarfraz et al., 2013).

4.1 Problem Definition

The problem that Optometer aims to solve is when a person has a still image from a CCTV system, which contains the depiction of a vehicle license plate where the registration number is illegible. The *scenario* is that an investigator has reviewed CCTV footage finding a vehicle of interest, taken a still image of that vehicle, cropped the image to just the license plate, and found the registration number to be illegible. This project constrained such illegibility to two factors, those being digital image resolution and optical quality. This project excludes other environmental factors such as orthogonality, illumination, and movement, for the purpose of bringing the scope of the project to within human and time resource budgets. Figure 1 below is an example of the problem. Figure 1 is a crop from an actual image of a vehicle license plate bearing registration number **1ABV346**. It is illegible on account of blurring caused by limits in optical quality, and lack of resolution.



Figure 1: Illegible Real-World Registration Number 1ABV346 Enlarged from 50px to 200px

4.2 Problem Significance & Complexity

A CCTV system capturing images of a vehicle license plate is unlikely to capture the license plate in ideal circumstances. Environmental variations such as illumination and background combine with plate variations such as occlusion, inclination, and allowable character sets (Duan, Du, Phuoc, & Hoang, 2005). The main focus of academic research in this area appears to be in ALPR systems. Du, Ibrahim, Shehata, and Badawy (2012) summarised the “four stages of an ALPR system” as image acquisition, license plate extraction, license plate segmentation, and character recognition. This project excludes the first step of the ALPR system process and assumes that the user has already extracted the license plate from the image. Subsequent to license plate extraction, ALPR systems segment the registration number into individual characters. Prior to actual segmentation is the transformation of the image to render it equivalent to orthogonal to the viewing plane. Once the cropped image is orthogonal, there are various methods by which systems can extract alphanumeric characters. Once character segmentation is complete, the task becomes one of Optical Character Recognition (OCR), which again is the subject of extensive research.

4.3 Existing Work & Approaches

The scenario constraining the problem domain to the last two of the four stage ALPR system process described by Du et al. (2012). As such this section of the report will review current technology and research in the areas of license plate segmentation and character recognition. In their review, Du et al. (2012) found five main methods of license plate segmentation.

The approach of Kanayama, Fujikawa, Fujimoto, and Horino (1991) involved finding vertical edges on a per-pixel basis in the image, in order to determine the boundary areas of each character in the registration number. Du et al. (2012) described this approach as simple and robust, but not robust against joined or broken characters. The approach of Duan et al. (2005) involved finding character boundaries by their contrast against the plate background colour. Du et al. (2012) described this method as more robust for character positioning but more affected by noise. The approach of Paliy, Turchenko, Koval, Sachenko, and Markowsky (2004) resizes all input images to a known template size, and makes assumptions about the geometry of registration number such as character dimensions, spacing and location. Du et al. (2012) described this technique as simple but not robust against differing plate geometry. The approach of Capar and Gokmen (2006) involved detecting alphanumeric characters by the shape of their outlines (contours). Du et al. (2012) described this approach as precise but slow. Lastly, the approach of Nomura, Yamanaka, Katai, Kawakami, and Shiose (2005) involved using algorithms that was able to merge fragments of characters and increase the visual thickness of the character for subsequent recognition. Du et al. (2012) described this approach as “more reliable” but “computationally complex”.

In summary, a review by Du et al. (2012) found that in ALPR research, there were five predominant methods for registration number segmentation, each with benefits and drawbacks. A review of these techniques found that each paper appeared to rely on input images of a resolution higher than those proposed in this project.

4.4 Results of Other Technologies

Existing technologies reviewed above in this report all appeared to hold an underlying assumption that there was enough resolution in the input image to conduct the stages of analysis. For example, Sarfraz et al. (2013) show the below (originally in Figure 7(a) of their paper) example of an extracted license plate image.



Figure 2: ALPR Extraction Sample (Sarfraz et al., 2013)

When comparing Figure 1 and Figure 2 (both above) there are a number of key differences immediately visible to the observer. Figure 2 has much greater optical clarity than Figure 1, and it appears from cursory inspection that the primary limiting factor of Figure 2 is image resolution. Naturally with better optical quality and enough digital resolution, the subsequent ALPR stages of segmentation and recognition become possible.

5.0 Optometer

An optometer is a device, used for measuring spherical and/or cylindrical prescription for eyeglasses, from the middle of the 18th century until around 1922, when modern instruments developed (Wikipedia 2019). The author chose this name because this project is about the application of synthetic lenses to an image to enhance clarity, which is analogous to the process of using an optometer.

5.1 Scope

Due to restraints of computing resources, time, and mathematical complexity, this project will restrict license plates to standard blue on white Western Australian (WA) license plates. At the time of this report, WA license plates had the following attributes shown in Table 1 below.

Table 1: Ruleset for Standard WA License plates

Index	Rule	Rationale
1	Numerical character "1"	As of 2019, license plates have yet to exceed "1"
2	Letter between "A" and "G"	As of 2019, license plates have yet to exceed "G"
3	Random letter	Any letter allowed
4	Random letter	Any letter allowed
5	Random number	Any number allowed
6	Random number	Any number allowed
7	Random number	Any number allowed

Although the standard WA license plate can have vanity text applied, this was outside the project scope as it increased the number of plate combinations and made distribution of character less even.

5.2 Technology Choice

The review of existing work and approaches in section 4.3 identified that most existing approaches worked on identifying individual characters through diverse methods. The approach taken by this project was therefore novel. The primary choice of technology for this project is the Super Resolution Generative Adversarial Network (SRGAN), which is a specific type of Generative Adversarial Network (GAN). This project chose a particular implementation of the SRGAN known as ESRGAN, by X. Wang et al. (2018). This section of the report provides a brief explanation of these technologies.

5.2.1 GAN

The Generative Adversarial Network (GAN) is a significant development in the field of machine learning. Goodfellow et al. (2014) developed the GAN, known then as *Generative Adversarial Nets* in 2014. The GAN is a system for generating a data model that involves the game theory concept of an adversarial process. A generative model G generates a data sample based on a training data, and a discriminative model D estimates whether the data sample from the training data (real) or whether the generative model G made it (fake). Iterations of this process improve both the generation and discrimination of data samples, toward the goal of producing data samples that the generative model made (fake) but difficult for an outside process to distinguish from real samples. Goodfellow et al. (2014) used the analogy of a team of counterfeiters (generative model) trying to produce fake currency while a police team (discriminative model) try to detect the counterfeits.

5.2.2 SRGAN

The Super Resolution Generative Adversarial Network (SRGAN) is a GAN adapted for use in image super-resolution. Image super-resolution is increasing the resolution of an image using inferred data from internal or external sources. Traditional image super-resolution techniques involve super-resolution using data inferred from the image itself (Dong, Loy, He, & Tang, 2015). SRGAN proposed conducting image super-resolution using a GAN trained with other images.

5.2.3 ESRGAN

ESRGAN is an implementation of a SRGAN by X. Wang et al. (2018). It is notable for having its source code publicly available and easily adoptable for the purposes of this project. X. Wang et al. (2018) also published BasicSR for public use, which is a set of tools for training ESRGAN models. ESRGAN aimed to improve SRGAN by employing a relativistic average GAN that used features before activation and used Residual-in-Residual Dense Blocks to make a deeper model.

A relativistic average GAN is one that contains a relativistic discriminator. A relativistic discriminator is where the discriminator “estimates the probability that the given real data is more realistic than a randomly sampled fake data” (Jolicoeur-Martineau, 2018). This differs from a basic SRGAN where the discriminator does not have any prior knowledge, and thus learns much more slowly. In practice this means that ESRGAN can learn at a useful rate over smaller data sets and with fewer iterations, which was well suited to the resource constraints of the project. Relativistic describes the discriminator judging whether an image is more realistic than another, rather than whether it is real or fake (X. Wang et al., 2018, p. 2).

Another feature of ESRGAN is the lack of Batch Normalisation (BN) layers. X. Wang et al. (2018) found that BN layers reduced the ability for the SRGAN to generalise and increased the probability of introducing artefacts.

Most notably, ESRGAN uses a basic block that X. Wang et al. (2018, p. 6) named the Residual-in-Residual Dense Block (RRDB). Macroscopically, the RRDB introduced blocks that had a greater number of connections between one another, which the authors noticed tended to improve performance. The term *Residual-in-Residual* refers to the fact that multiple levels of the learning structure employed residual learning, not just one, so the name describes nested residual learning.

Overall, ESRGAN was a suitable choice for this project due the availability of its code, and its ability to produce high quality super-resolution images. Technologically ESRGAN is an implementation that tends to be very eager to learn and requires less training than other SRGAN systems. This, combined with the availability of some pre-trained models, made ESRGAN the sensible choice of system upon which to base this project.

5.3 Dataset Generation

The first part of this project was to generate a series of random license plate images that made a modest representation of the given dataset.

5.3.1 Image Template

The author used the following stock image, sourced from [plateshack.com](https://www.plateshack.com). The author used Adobe Photoshop to generate a blank template, with the intention of using it as a background for randomly generated alphanumeric characters.

Table 2: Stock & Blank License plate



5.3.2 Generating a Dataset

The author created a command line Python program that generated random allowable license plate combinations. Table 1 above shows the ruleset that the Python program implemented to create random allowable standard WA license plates.

At this early stage of the project, the author restricted plate generation to standard metropolitan Perth plates. This allowed more rapid prototyping of the system and progressed the workflow instead of spending time creating a forensic representation of every series of plates possible. The author decided that the project could expand scope into a more exact representation of all allowable plates if it would help training.

Table 3: Sample of Random Metropolitan Registrations

1GYE896	1EER016	1GVI787	1GXN403	1GTP826	1DBC841	1DPI901	1CZF599
1DYH507	1AXX417	1EVA931	1ASV515	1DVS850	1DXU899	1DSN022	1CTU930
1CEY103	1GSJ376	1ACE715	1DSM571	1GNN969	1BKS451	1ERG876	1DEV845
1AOY969	1DFD121	1ALO683	1DEV845	1DPV473	1GYT465	1BVN575	1DDM891

5.3.3 Typeface Selection

Another task needed for the generation of synthetic license plate images was for the author to find a font that was an exact representation of the original used by the Western Australian Department of Transport. The author submitted the stock image to fontquirrel.com/matcherator and found the typeface *Dancing Juice & Salabat*, created by Dondon Nillo, to be a fair match from the selection of free typefaces. Table 4 below shows the author's rendition of the stock image using a typeface with valid license for use in this project.

Table 4: First Synthetic License plate



Table 4 above shows that *Dancing Juice & Salabat* effectively captures the spur and tail of the number "1" and is a fair approximation of the counter in "A". "Z" and "S" are almost identical. In this example, "9" and "2" differ slightly, but the author found this margin of error to be acceptable given that this typeface was macroscopically like the original. During this stage of the project, the author decided against including the printer's ornament. The rationale for this decision was that the printer's ornament has no plaintext equivalent because it is a Unicode character. An alternative approach of dividing the template into two sections proved unworkable because Adobe Photoshop could only apply one dataset to an image at a time.

Table 5: Adobe Photoshop Font Settings

Variable	Setting
Typeface	Dancing Juice & Salabat
Weight	Regular
Size	320pt
Kerning	Optical Mode
Vertical Scale	125%
Horizontal Scale	125%
Colour	365bd0

5.3.4 Image Synthesis

The author imported the random dataset into Adobe Photoshop and used it to generate a series of images using the base synthetic image from Table 4 above. Table 6 below is a sample of images generated using the author's template, and using a dataset of

Table 6: Synthesised Random Images





5.3.5 Improvements to Synthesis

Two issues found with the first batch of synthetic images was that because the chosen typeface is not a monospaced font, the overall length of the generated characters is variable. Table 6 above shows how some character sets leave more blank space at the right side of the image, where others do not. The original image shows that this is not the case for real license plates.

Table 7: Improved Synthetic Plate



The primary improvements shown in Table 7 above are that the character set is justified between the bounds of the original character set geometry. Again, the lack of printer's ornament caused issues because the kerning of the improved synthetic image was still much more than the original due to having to span the entire width of the character field.

Table 8: Improved Adobe Photoshop Font Settings

Variable	Setting
Typeface	Dancing Juice & Salabat
Weight	Regular
Size	320pt
Kerning	Optical Mode
Vertical Scale	120%
Horizontal Scale	125%
Colour	365bd0

Table 9: Adobe Photoshop Emboss Settings

Variable	Setting
Style	Emboss
Technique	Chisel Soft
Depth	1%
Direction	Up
Size	4px
Soften	8px
Angle	90°
Altitude	5°
Shadow Opacity	60%

Table 10: Sample of Improved Synthetic Plates



Table 10 above shows a sample of the improved synthetic plates. The overall colouring appears more authentic, and the justified text gets rid of the whitespace found at the right of the first synthetic plates. The plates in Table 10 above were sufficiently analogous to the original for use in training. The author used variable substitution in Adobe Photoshop to import random samples of standard plate character sets. The author generated 50000 images using this method.

5.3.6 Dataset Preparation

Once Photoshop had exported 50000 PSD images, the next task was to prepare the dataset for input to the training system. To prepare the images, the author used Irfan View to resample the images to a high resolution of 200 pixels width, and a low resolution of 50 pixels width. All the images went into a single folder, causing Irfan View to skip duplicates automatically. This resulted in a total of 45178 unique images.

Table 11: Method for Image Resampling

Variable	Original	High Resolution	Low Resolution
Width	695	200	50
Height	249	72	18
File Type	PSD	PNG	PNG
Resize Method	N/A	Lanczos	Lanczos

The author also used Linux shell commands to make a listing of all of the unique images in alphabetical order, so that with the images being named according to their character set, for example 1ABC123.png, there would be many ways that other researchers could evaluate any image for what characters it should contain.

5.4 Adaptation of BasicSR

With the above procedurally generated plate sets, the next task was to train the system with them. The intention of this project was to train [ESRGAN](#) using this dataset, so the author found the [BasicSR](#) repository, which had the tools needed to train a model for ESRGAN. To train BasicSR, the training dataset had to have low resolution images four times smaller than high resolution images, and about ten percent of image pairs left over for evaluation.

The author developed a fork of BasicSR for use in this project. Primary modifications include a top-level bash script to manage execution of the various Python files. Due to its author replacing BasicSR with [MMSR](#), BasicSR lost some compatibility with updated dependencies. The author estimated that BasicSR was simpler than MMSR and would be easier to change to suit this project. BasicSR was slightly out of date and did not work instantaneously upon download. To remedy this, the author's version changed program code for compatibility with the latest versions of Nvidia graphics card drivers, Nvidia CUDA, and the Nvidia CUDA toolkit.

Machine learning is a compute intensive application and prior to commencement of training it was prudent to make estimations of the processing power at hand in comparison to those used in the creation of known datasets. Wang, the author of ESRGAN and BasicSR, mentioned in [a forum comment](#), that training the well-known dataset RRDB_PSNR_x4 took about one week using an Nvidia Titan XP graphics card. This singular data point was the basis of data estimations for this project. The Nvidia Titan XP graphics card is known to be much better performing than the Nvidia Quadro P400 available for this project. Table 12 below shows primary performance metrics of the two devices and a summary of the differences.

Table 12: Performance Comparison of Nvidia Titan XP and Nvidia Quadro P400

Feature	Titan XP	Quadro P400	Summary of P400
Memory Bus	384 bit	64 bit	6x Less
Memory Size	12GB	2GB	6x Less
Memory Bandwidth	547.6 GB/s	32.10 GB/s	17x Less
Effective Memory Clock	11408 MHz	4012 MHz	2.8x Less
Floating Point Performance	12.15 TFLOPS	641.0 GFLOPS	18x Less

From the results of Table 12 above, the author estimated that training a comparable replacement to RRDB_PSNR_x4 would take at best 6 weeks, and at worst 18 weeks. This was not practical for the timeline of this project. BasicSR did however, give the possibility of using a pretrained model as a base that further training could improve. With this decision made, the next step was to execute some training to test the training system.

Table 13: Results for Changing Thread Count in 1000 Iterations

Number of Threads	Batch Size	Time for 10 Iterations
1	1	19.425s
2	1	17.754s
3	1	18.320s
4	1	17.371s
5	1	18.096s
6	1	18.489s
6	2	23.033s
6	3	26.773s

Table 13 above indicated that an increasing thread count tended to increase the time taken per iteration to a very small extent, and that increasing the batch count tended to increase the time taken per iteration by a more significant extent. These values provided input for the selection of final resource usage values in the training of models as part of experimentation.

Table 14: BasicSR Configuration Changes

Variable	Code	Default	Modified
Number of Threads per GPU	n_workers	6	4
Batch Size	batch_size	16	1

Through experimentation in the test environment, the author determined that the optimal resource usage variables for BasicSR were as in Table 13 above indicated that an increasing thread count tended to increase the time taken per iteration to a very small extent, and that increasing the batch count tended to increase the time taken per iteration by a more significant extent. These values provided input for the selection of final resource usage values in the training of models as part of experimentation.

Table 14 above, where both the thread count and batch size was reduced to accommodate the limited computing resources used for the majority of this project.

6.0 Method

With BasicSR and ESRGAN configured for the project, it came to the method phase to determine how to conduct repeatable and useful experiments to improve the pre-trained models with the supplied synthetic data.

6.1 Model Comparison

Prior to conducting further training, the author first prepared a set of 100 license plates that the evaluation metrics would use across all changed variables. This was essential to enable the tests and experiments to be repeatable. Firstly, the author calculated the average PSNR and SSIM of the 100 license plates using two pre-trained models, as shown in Table 15 below. Given the names of the models, it was unsurprising that the model trained for PSNR achieved a better PSNR than its rival. The PSNR model also exhibited improved average SSIM over the 100 license plates when compared against the alternative. The PSNR model also had better legibility with less introduced visual noise. This led to the PSNR model's choice as the base model, which the experiments would subject to further training.

Table 15: Metrics of Pretrained Models

Model	Mean PSNR	Mean SSIM
RRDB_ESRGAN_x4.pth	13.952684	0.190697
RRDB_PSNR_x4.pth	14.231533	0.232456

6.2 Increasing Iterations

The first series of tests involved changing the number of iterations over two datasets, one of 1000 training and 100 validation images, and the other of 45178 training and 4524 validation images. The aim of having two series was firstly, to estimate the impact of substantial amounts of file input and output on performance, and to determine whether the sample size influences the accuracy of the resultant models.

The author of ESRGAN recommended 400000 iterations over the pre-trained models for substantial effect on the models, but to keep methods scientific, the author began at a much lower threshold of 1000 iterations, increasing by a factor of 2 to 128000. The author selected a factor of two to balance time constraints with the need to sample at a range of magnitudes for measurement of effect. The final iteration counts were 1000, 2000, 4000, 8000, 16000, 32000, 64000, and 128000.

6.3 Metrics

The goal of this project was to increase the legibility of license plates, and this report here acknowledges the subjective nature of legibility. Where possible, this report provides sample images so that the reader may judge legibility of results. To supplement this subjective measure, this project also analysed all results according to two popular image quality metrics, PSNR and SSIM (Hore & Ziou, 2010). These metrics are popular in the field of SRGAN development because they provide a simple, single number representation of the accuracy of a process.

6.3.1 PSNR

The Peak Signal to Noise Ratio (PSNR) measures losses introduced by compression distortion in images (Salomon & Motta, 2010) by measuring the maximum signal strength divided by the maximum strength of signal noise. It is agreed that PSNR does not equate to human perception of errors and quality, but it is still an accepted metric in this field (Salomon & Motta, 2010).

Equation 1: Mean Square of Errors (MSE)

$$MSE = \frac{1}{n} \sum_{i=1}^n (P_i - Q_i)^2$$

Equation 2: Peak Signal to Noise Ratio (PSNR)

$$PSNR = 20 \log_{10} \frac{\max_i |P_i|}{\sqrt{MSE}}$$

Equation 1 and Equation 2 above show that the PSNR is calculated by first calculating the mean of the square of errors (MSE) of the image on a per-pixel basis, and then using its root as the denominator under the maximum signal strength for a value that is then converted to a logarithmic scale. The resultant number has unit decibel (dB). Absolute values of PSNR are meaningless (Salomon & Motta, 2010, p. 464) and on account of this limitation, this report does not give any particular target for this metric, instead using it only as a comparative value.

6.3.2 SSIM

Structural Similarity (SSIM) was the second method used in this project to objectively measure output files against ground truth to give an indication to the quality of trained models and effect of variables.

Equation 3: Structural Similarity (SSIM)

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)}$$

SSIM was originally developed by Z. Wang, Bovik, Sheikh, and Simoncelli (2004) as an improvement over PSNR, which they noted did not correlate particularly well with human perception. SSIM is effective because it considers and compares the luminance, contrast, and structure of two images. A value of 0 reflects two images with no correlation, while a value of 1 reflects two identical images. Z. Wang et al. (2004) suggested that SSIM should not be the single measure of image quality loss but exist alongside other measurements such as MSE and PSNR. This report considers the limitations the value and limitations of SSIM by always measuring it alongside PSNR, as well as comparing it against the subjective visual assessment.

6.3.3 Improvement Score

Equation 4: Improvement Score Calculator

$$\text{Improvement (\%)} = \left[\left(\frac{\text{Trained Model Score}}{\text{Pre - Trained Model Score}} \right) - 1 \right] \times 100$$

In addition to the quantitative results that this report describes in sections 6.3.1 and 6.3.2 this report also uses a common percentage *improvement score* to give an easily understandable number representing the differences between a particular result, and the results of the pre-trained model over the same measure. There is limited value in the improvement score, especially since the improvement score in PSNR are not to the same scale as those of SSIM. Percentage improvements on a logarithmic scale are inevitably much smaller than those on a decimal scale. The improvement score is, therefore, better suited to SSIM, but again this is a purely relative measure on account of PSNR and SSIM being entirely comparative numbers.

7.0 Results

This section of the results tabulates the outcomes of the method explained previously and provides graphs to indicate data trends. This section encompasses all tests carried out during development of the software systems, the decisions, and their rationale.

Table 16: Ground Truth and Low-Resolution Samples

Type	Sample
Ground Truth	
Low Resolution	
Real-World	

As Table 16 above shows, the ground truth images were subject to down sampling to 50 pixels in width, and then further subject to two rounds of a blur filter being applied. This made the low-resolution images analogous to the real-world image sample shown in the table. The real-world image is taken from Figure 1 on page 5 of this report.

7.1.1 Iterations Over 1000 Images

Sample images are in Appendix 1: Sample Image of Iteration Tests over 1000 Training Images.

Table 17: Control Variables for Increasing Iterations

Control Variable	Value
Training Images	1000
Validation Images	100
Evaluation Images	100
Pretrained Model	RRDB_PSNR_x4

Table 18: Results for Increasing Iterations over 1000 Images

Iterations	Mean PSNR	Improvement	Mean SSIM	Improvement
0	14.231533	N/A	0.232456	N/A
1000	14.213339	-0.13%	0.246539	6.06%
2000	14.180878	-0.36%	0.238300	2.51%
4000	14.219912	-0.08%	0.239765	3.14%
8000	14.085603	-1.03%	0.248589	6.94%
16000	14.255639	0.17%	0.254330	9.41%
32000	14.309129	0.55%	0.256638	10.40%
64000	14.145575	-0.60%	0.251079	8.01%
128000	14.273890	0.30%	0.251808	8.33%

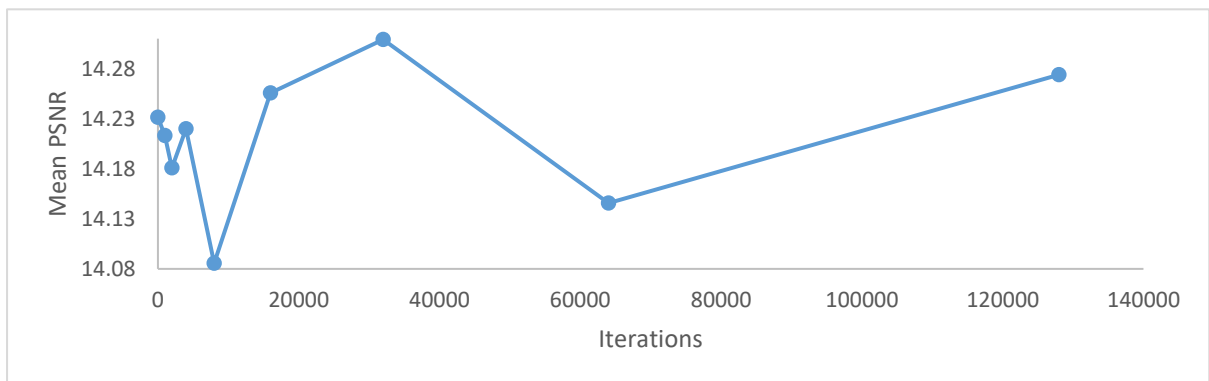


Figure 3: Mean PSNR for Iterations over 1000 Images

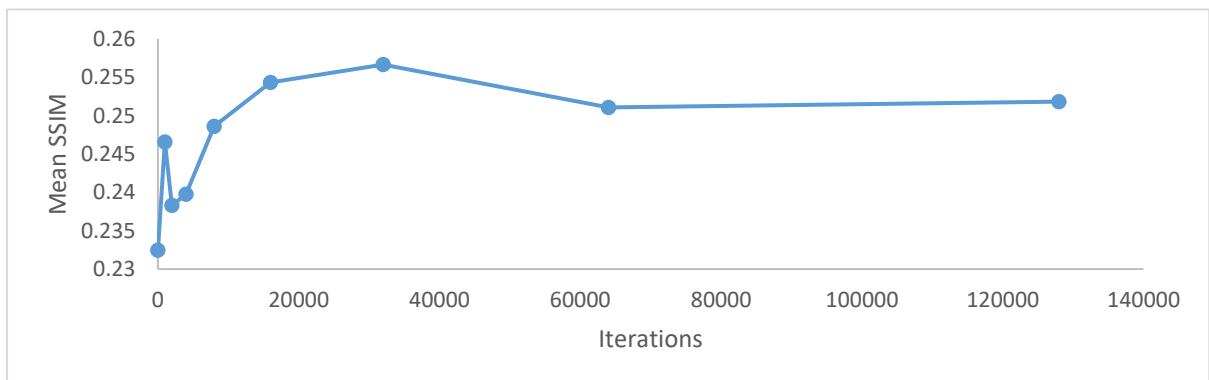


Figure 4: Mean SSIM for Iterations over 1000 Images

7.1.2 Iterations Over 45178 Images

Sample images are in Appendix 2: Sample Image of Iteration Tests over 45178 Training Images.

Table 19: Control Variables for Increasing Iterations

Control Variable	Value
Training Images	45178
Validation Images	4524
Evaluation Images	100
Pretrained Model	RRDB_PSNR_x4

Table 20: Results for Increasing Iterations over 45178 Images

Iterations	Mean PSNR	Improvement	Mean SSIM	Improvement
0	14.231533	N/A	0.232456	N/A
1000	14.202454	-0.20%	0.244970	5.38%
2000	14.169065	-0.44%	0.249539	7.35%
4000	14.228152	-0.02%	0.247098	6.30%
8000	14.106782	-0.88%	0.252167	8.48%
16000	14.183480	-0.34%	0.253738	9.16%
32000	14.215105	-0.12%	0.261269	12.40%
64000	14.287242	0.39%	0.254009	9.27%
128000	14.242665	0.08%	0.255075	9.73%

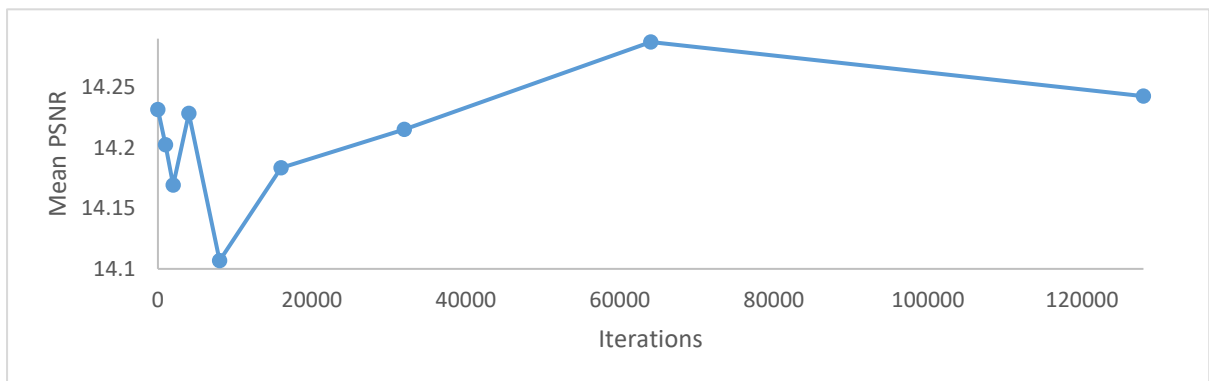


Figure 5: Mean PSNR for Iterations over 45178 Images

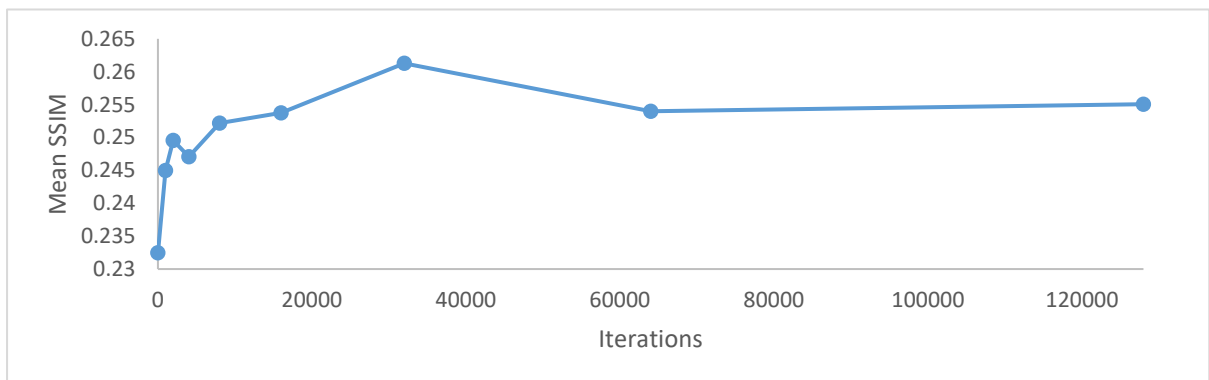


Figure 6: Mean SSIM for Iterations over 45178 Images

7.1.3 Learning Rate for 1000 Images and 1000 Iterations

Sample images are in Appendix 3: Sample Images of Learning Rate Tests of 1000 Images over 1000 Iterations

Table 21: Control Variables for Increasing Learning Rate

Control Variable	Value
Training Images	1000
Validation Images	100
Evaluation Images	100
Pretrained Model	RRDB_PSNR_x4
Iterations	1000

Table 22: Results for Learning Rate over 1000 Images

Learning Rate	Mean PSNR	Improvement	Mean SSIM	Improvement
0.0001 (Default)	14.101473	N/A	0.246677	N/A
0.0002	14.061443	-0.28%	0.251790	2.07%
0.0004	13.430871	-4.76%	0.256004	3.78%
0.0008	5.837994	-58.60%	0.122854	-50.20%
0.0016	3.786857	-73.15%	0.037608	-84.75%

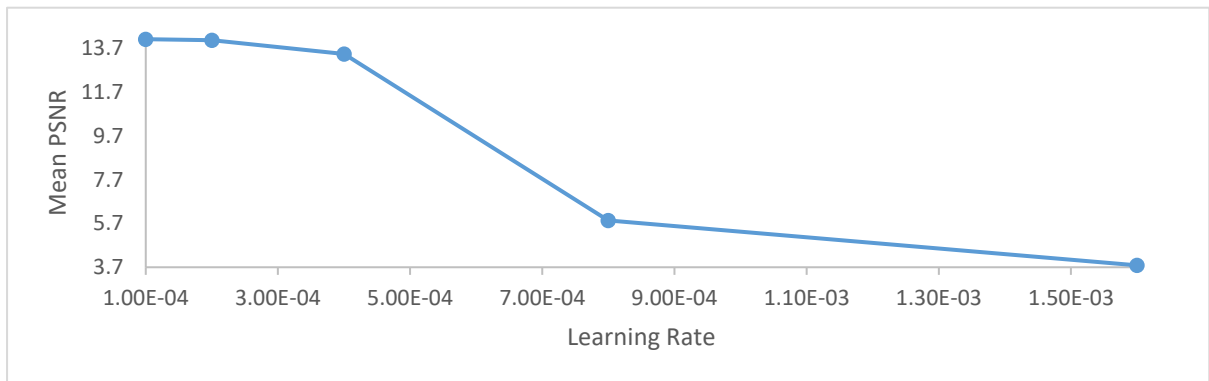


Figure 7: Mean PSNR for Learning Rate over 1000 Images

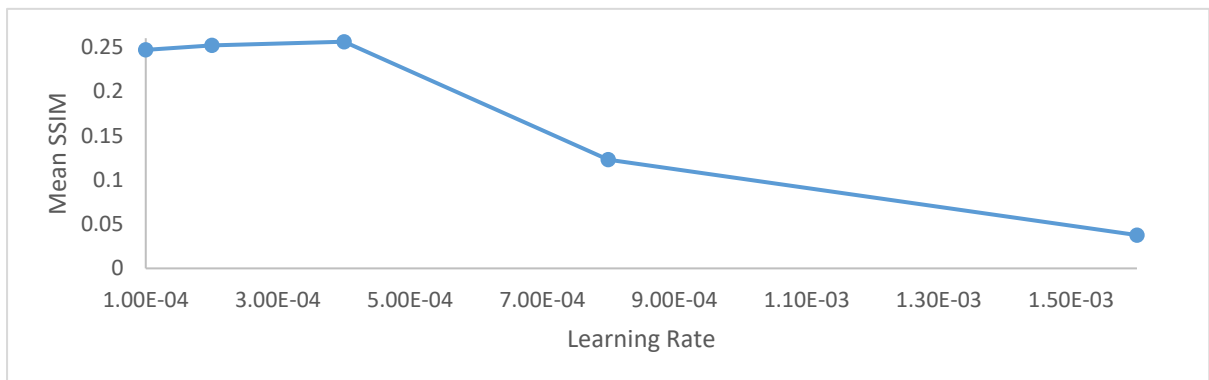


Figure 8: Mean SSIM for Learning Rate over 1000 Images

7.1.4 Iterations for 1000 Images at 0.0004 Learning Rate

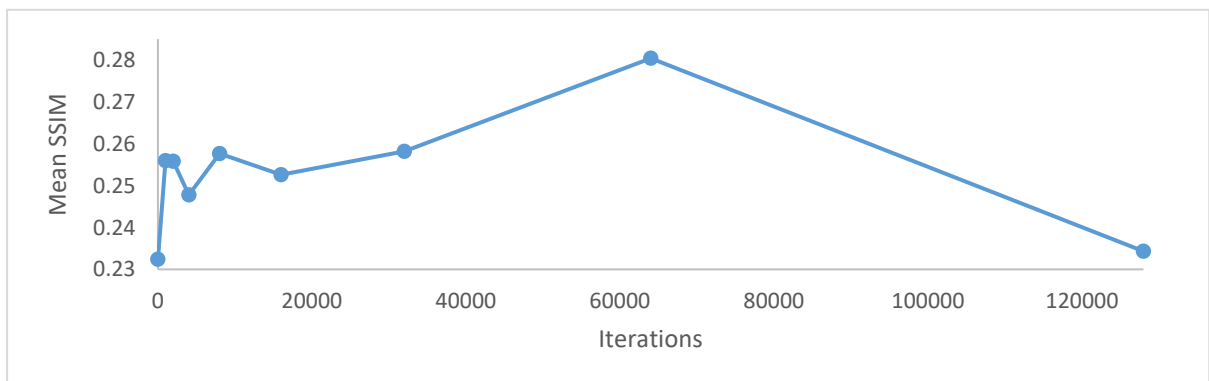
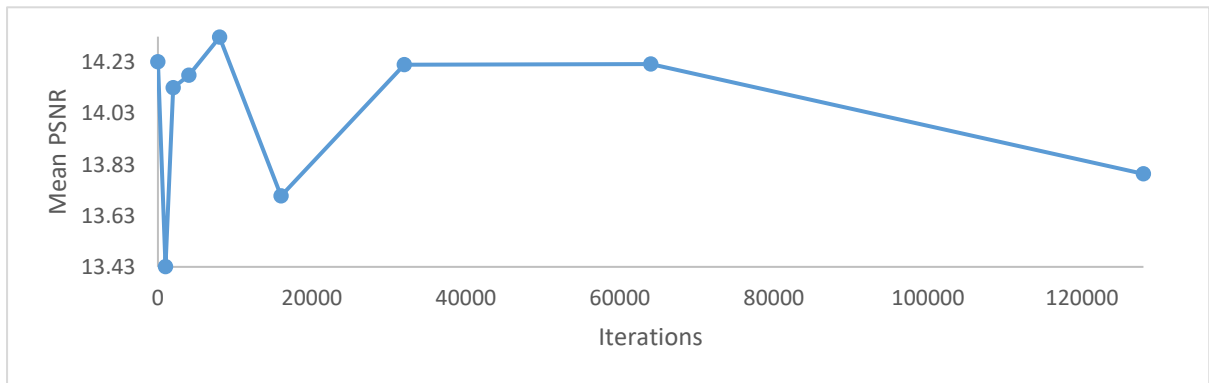
Sample images are in Appendix 4: Sample Images of Iteration Tests for 1000 Images and 0.0004 Learning Rate.

Table 23: Control Variables for Increasing Iterations at 0.0004 Learning Rate

Control Variable	Value
Training Images	1000
Validation Images	100
Evaluation Images	100
Pretrained Model	RRDB_PSNR_x4
Learning Rate (Generator and Discriminator)	0.0004

Table 24: Results for Increasing Iterations over 45178 Images at 0.0004 Learning Rate

Iterations	Mean PSNR	Improvement	Mean SSIM	Improvement
0	14.231533	N/A	0.232456	N/A
1000	13.430871	-5.63%	0.256004	10.13%
2000	14.130483	-0.71%	0.255801	10.04%
4000	14.179063	-0.37%	0.247820	6.61%
8000	14.327632	0.68%	0.257626	10.83%
16000	13.706979	-3.69%	0.252617	8.67%
32000	14.220140	-0.08%	0.258228	11.09%
64000	14.223097	-0.06%	0.280424	20.64%
128000	13.793866	-3.08%	0.234346	0.81%



7.1.5 Effect of Input Image Degradation

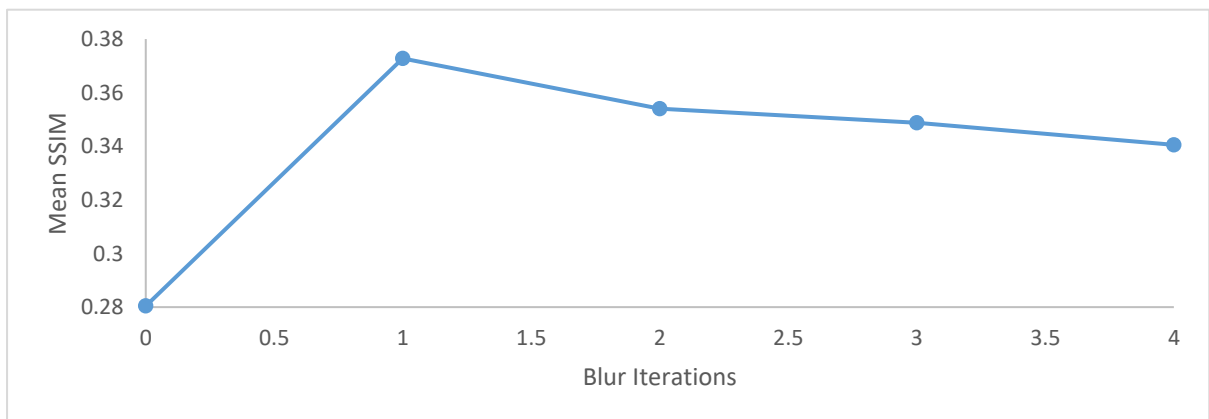
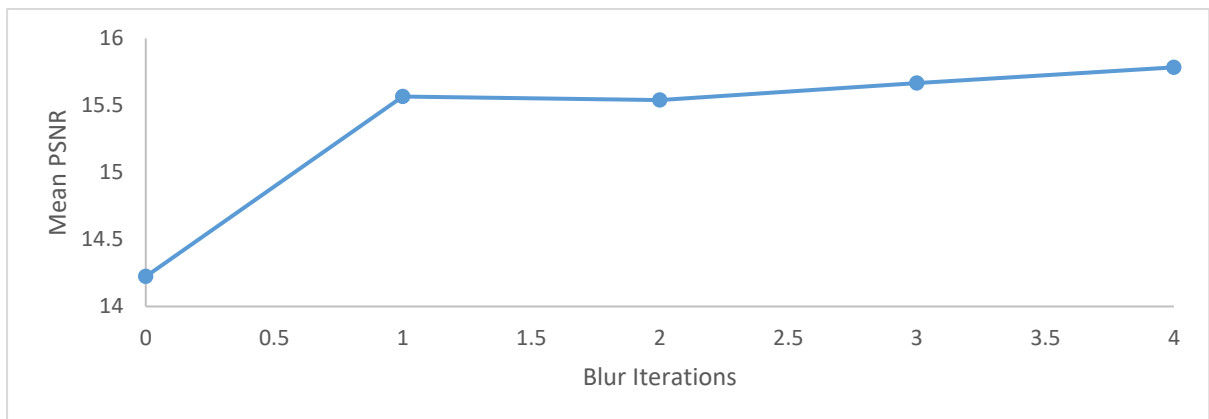
Sample images are in Appendix 5: Quality Comparison as Input Image Degrades.

Table 25: Control Variables Input Image Degradation Tests

Control Variable	Value
Model	RRDB_PSNR_x4.pth
Training Images	1000
Testing Images	100
Pretrained Model	RRDB_PSNR_x4
Learning Rate (Generator and Discriminator)	0.0004
Iterations	64000

Table 26: Results for Control Variable Input Image Degradation

Blur Iterations	Mean PSNR	Improvement	Mean SSIM	Improvement
0	14.223097	N/A	0.280424	N/A
1	15.566439	9.44%	0.372767	32.93%
2	15.538547	9.25%	0.354019	26.24%
3	15.666177	10.15%	0.348821	24.39%
4	15.783697	10.97%	0.340570	21.45%



8.0 Discussion

Testing revealed several interesting trends in training behaviour, as well as both qualitative and quantitative outcomes. Qualitatively, most training of the pre-trained model resulted in super-resolution output images that were visually more authentic to the original ground truth variables. This section precedes a quantitative analysis of the result measures.

Table 27: Sample Comparison of Results

Resize	Pretrained PSNR Model
	
PSNR: 14.284954 SSIM: 0.207648	PSNR: 14.231533 SSIM: 0.232456
Lanczos Resample	Optometer Best Result
	
PSNR: 14.420582 SSIM: 0.239977	PSNR: 14.223097 SSIM: 0.280424

Referring to Table 27 above, the author observed that the best efforts of training in this project resulted in an output sample image that was visually much sharper than non-machine learning techniques such as the Lanczos Algorithm. The author also observed that subjectively the training had a positive result on the clarity of the output image compared to the pre-trained model.

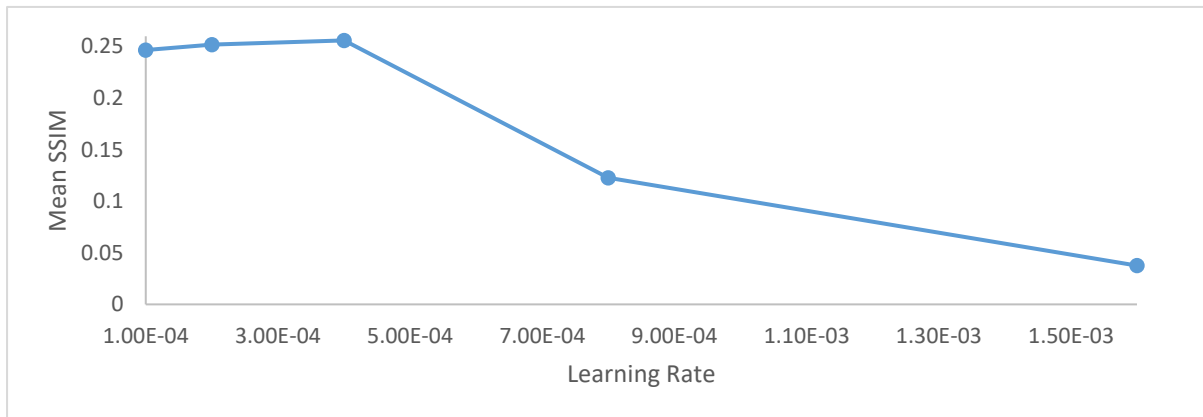
8.1 Effect of Reduced Input Image Quality

Reducing the image quality of the input for the trained model resulted in a quantitative increase in both PSNR and SSIM values. This suggested that the trained model was robust against degradation in image quality of the input. When viewing the images qualitatively however, it became clear that there was a threshold below which the increased PSNR and SSIM ceased to reflect legibility. These positive quantitative results only suggested at the trained model's ability to deal with degraded image quality.

8.2 Effect of Learning Rate

Of the two major variables altered during experiments was the learning rate of the ESRGAN algorithm. Figure 9 below shows the effect that altering the learning rate had on the mean SSIM of output models. Figure 9 refers to the results of section 7.1.3 on page 18. The range of the independent variable of learning rate, was between 0.0001 and 0.0016. The minimum number was the supplied learning rate of ESRGAN, and 0.0016 was the point at which the author established that the mean SSIM was dropping to values that lacked any experimental value.

Figure 9: Effect of Learning Rate on Mean SSIM



At a learning rate of 0.0004 there was a local maximum of 0.256004 in the mean SSIM measure of model performance. This represented a 3.78% improvement of SSIM value compared to the pre-trained model value of 0.246677. After the learning rate of 0.0004, there was a heave decrease in SSIM and PSNR values, which indicated a ceiling to the useful training rate somewhere near or just after 0.0004. Larger values of training rate resulted in images that ceased to have any resemblance to the original.

Table 28: Limit of Useful Learning Rate Values


0.0004	0.0008
	

Table 28 above shows how above this 0.0004 learning rate value, the model produced markedly greater error values, shown both visually in the resultant image, as well as the PSNR and SSIM measure. Above the 0.0004 training rate, the training collapses the model into a much less sophisticated system. The image on the right in the table above shows that at a learning rate of 0.0008, the model trained on 1000 iterations of 1000 images loses the ability to contribute valuable data for super-resolution. The primary colour palate is possibly a mean value for the colour palate of the original, and the areas of primary colour contrast are concentrated on areas of the image where there was the most consistent features, such as the first digit of the license plate character set, and the yellow sun logo. This result demonstrated that when the learning rate increased too far, the model collapsed. This appears to be a case of where the discriminator learning rate is too high, training will stop (Jolicoeur-Martineau, 2018, p. 2).

8.3 Effect of Increasing Iterations

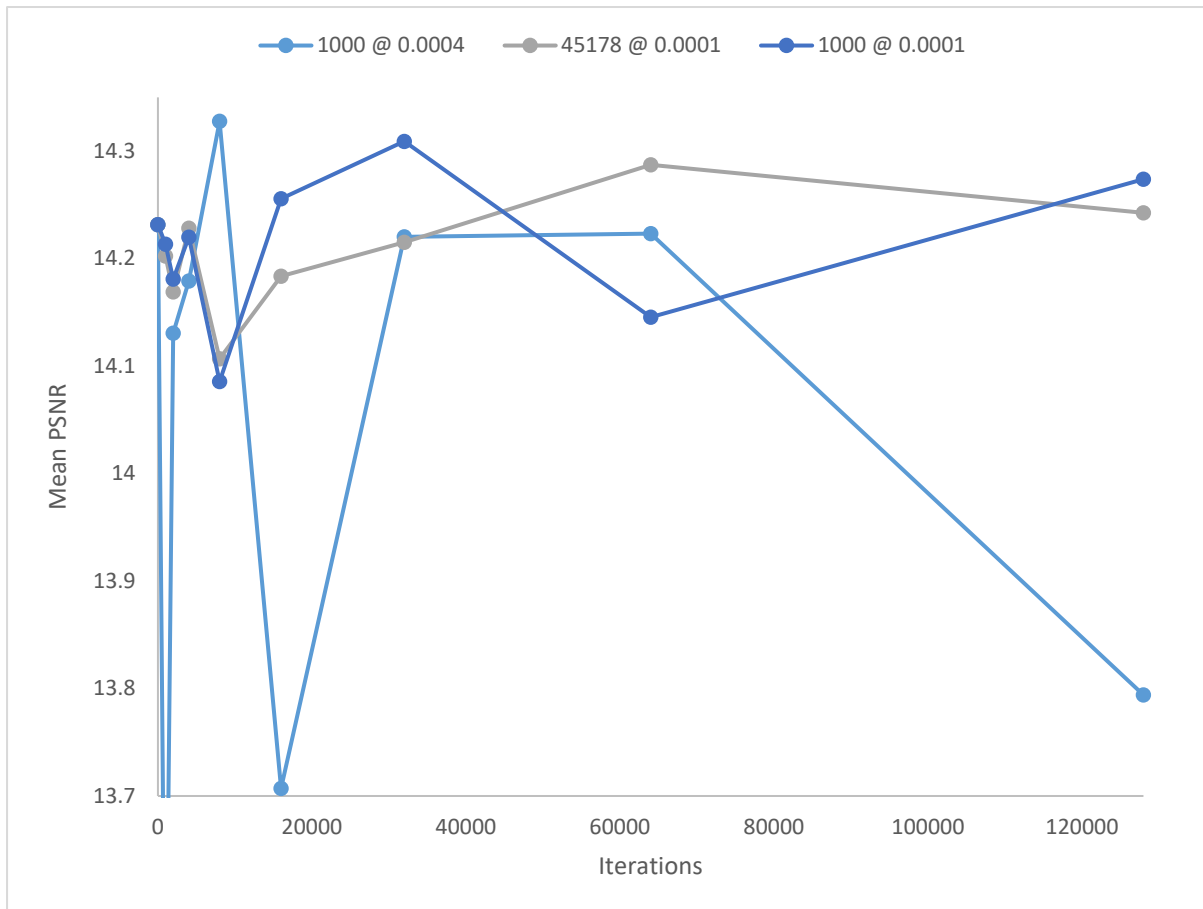


Figure 10: Mean PSNR Comparison

Figure 10 above is a graph of all mean PSNR results for test series involving increasing the number of iterations. The graph shows that at the default learning rate of 0.0001, there is insignificant effect in increasing the number of sample images from 1000 to 45178, because although there are minor differences, the overall trends in the results are similar. More interestingly, the increased learning rate of 0.0004 produced more variable results in the PSNR values across the iteration range. For this series, there were two large minima at 1000 and 16000 iterations, where the trained model exhibited much worse PSNR performance compared to other iteration values. Between 32000 and 64000 iterations all test series had similar performance. The test series at learning rate 0.0001 showed promise that further training beyond 128000 iterations might be worthwhile future work. On the other hand, the 0.0004 learning rate series appeared to drop noticeably at larger iteration values. This is suggestive that at the higher learning rate the model may exhibit over-training across the training data set.

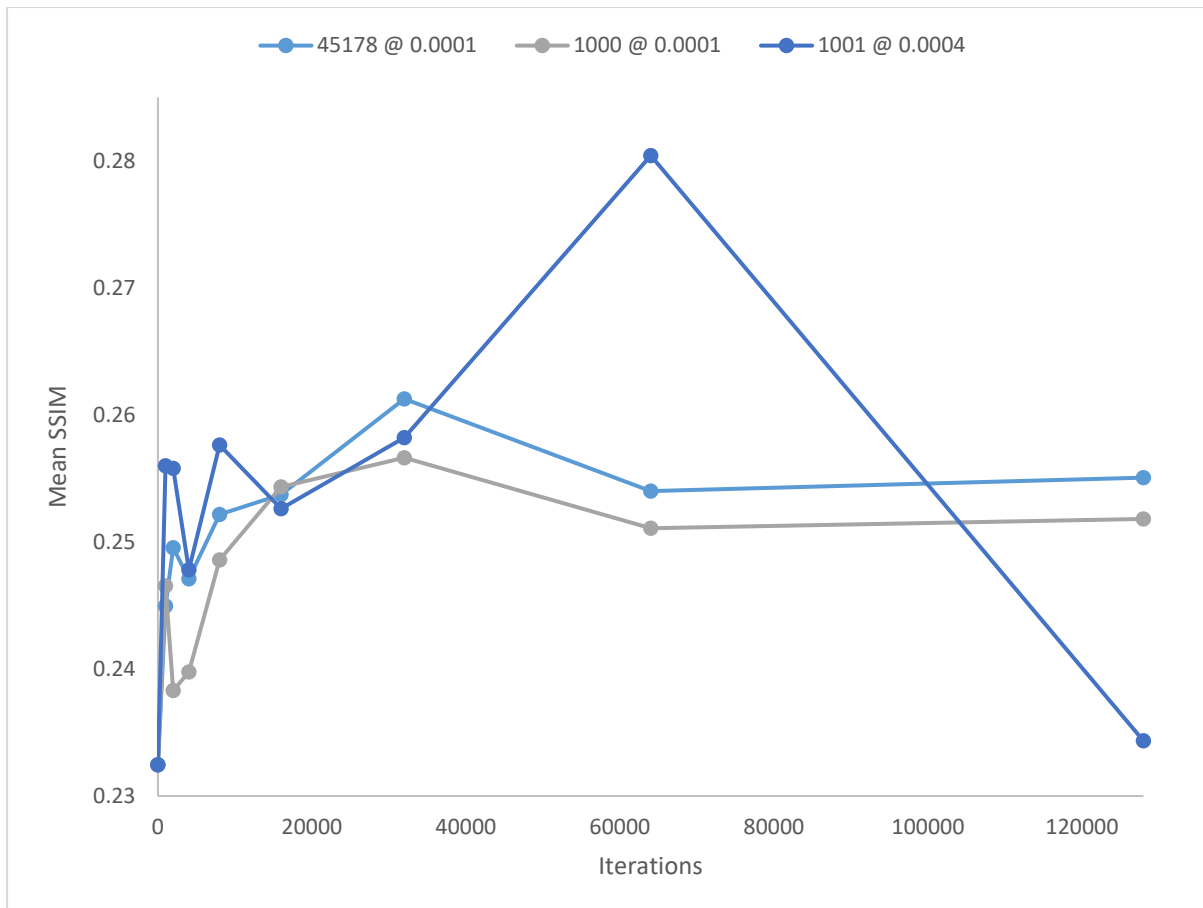


Figure 11: Mean SSIM Comparison

In Figure 11 above there is slightly different information to be gained. As with Figure 10 previously, the results at a learning rate of 0.0001 show that dataset size does not appear to be a significant contributor to model performance. In both cases, the results suggest strong potential for further iterations to give greater improvements, and this would be a worthwhile direction for future research. The results for increased learning rate at 0.0004 produce a substantial improvement to the model's SSIM performance at 64000 iterations, and it was this set of training variables that produce the best result of this project. Notably, the SSIM again reduces at higher iteration counts, just like it did with PSNR. This corroborates the idea that there may be a threshold above which the model starts to become over trained. It is also possible that this local maximum threshold is subject to other experiment constraints. It would be useful to establish whether data set size affected this local maximum.

The most positive result from the experiments was 64000 iterations over 1000 registration plates at a learning rate of 0.0004. Although this did not particularly affect PSNR (-0.06% improvement) it yielded a 20.64% increase in the SSIM numerical value compared to the value of the pre-trained model. Treating the SSIM as a percentage, this configuration yielded a SSIM value that was 4.79% closer to the ground truth value than the pre-trained model.

8.4 Shortfall of Synthetic Data

Table 29: Quantitative Analysis of Synthetic Data

Original	Improved Synthetic
	
PSNR: ∞ SSIM: 1	PSNR: 13.464366 SSIM: 0.594334

Table 29 above shows a samples of an actual registration plate (on the left) and the final output of the synthetic registration plate generation system devised for this project. As previously, the author applied the two measures of PSNR and SSIM to a scaled version of the synthetic image. Although the PSNR is not promising, the SSIM more closely resembles a visual analysis, showing that although there are some noticeable changes in the quality of the lettering, the synthetic registration plate is still 59.43% of the original according to the SSIM as a percentage.

Overall, the techniques employed in this project yielded results far below that of the synthetic registration plate. The main challenge is to reduce the shortfall between the super-resolution outputs and the ground truth. In this case the synthetic data demonstrated the usefulness of highly specific datasets in the problem domain. Inevitably authentic training data would be preferable for real-world applications, but these endeavours were outside the scope of this project.

Possible further work in this field could involve obtaining permission to use official fonts and liaison with image manipulation experts to yield a more accurate rendition of the registration plate for procedural generation. There is also the element to which the textual data forming the basis for the training set might be re-examined to determine its shortfalls. Specifically with Figure 12 below there is a suggestion that the lack of variance in the first digit 1 may have resulted in overtraining of that particular area of the SRGAN model. Visually, the 1 is noticeable faded compared to the contrast of other characters that are enhanced subject to greater variation of training data.



Figure 12: Best Super-Resolution Sample

It would be worth dividing the registration plate vertically and determining which areas of the image have better or worse PSNR/SSIM compared the original, to determine whether having a small but well known set of characters (such as the first number and first letter) helped, or in fact hindered the system from correctly conducting super-resolution on the images.

8.5 Effects of Lossy Input

At this time, it is relevant to mention a decision that the author made during the project that negatively impacted the visual acuity of the results for the cause of realism.

Table 30: Comparison of Model Performance on JPG and PNG inputs

File Type	JPG	PNG
Ground Truth		
Low Resolution		
Lanczos		
Base Model		
Optometer Best		

Based on the problem scenario, the author chose to use lossy low-resolution input images in JPG format to train and test the system. Table 30 above shows the differences in performance between the JPG process pipeline used in this project and the possible results using a lossless low-resolution image in a PNG format. The legibility of number plates super-resolved from a PNG base had better results because they had much less signal loss to have to restore. Having low-resolution images that a system had perfectly down sampled with no loss in quality bore no relation to the scenario, where a low quality lossy low-resolution image would be the subject.

Table 30 above shows that a properly trained ESRGAN system can be effectively adapted to a specific problem domain such as this project, as demonstrated by the Optometer Best sample image, which exceeds any other super-resolution method tested. Observing the images resolved from lossless low-resolution samples, while visually impressive, does not reflect the reality where input images from a CCTV system would always be lossy, such as the JPG images used in this project. Nonetheless, the PNG results demonstrate the malleability and adaptability of ESRGAN models within the testing range conducted during this project.

8.6 Comparison to Other Published Work

Scholarly searches for other research in this problem domain identified a paper on the same topic by Tian, Yap, and He (2012). A key difference with the work of Tian et al. (2012) is that their research included scope for an Optical Character Recognition (OCR) component, which this project did not. The inclusion of an OCR component in the training framework allowed their work to achieve much higher PSNR values for the result.

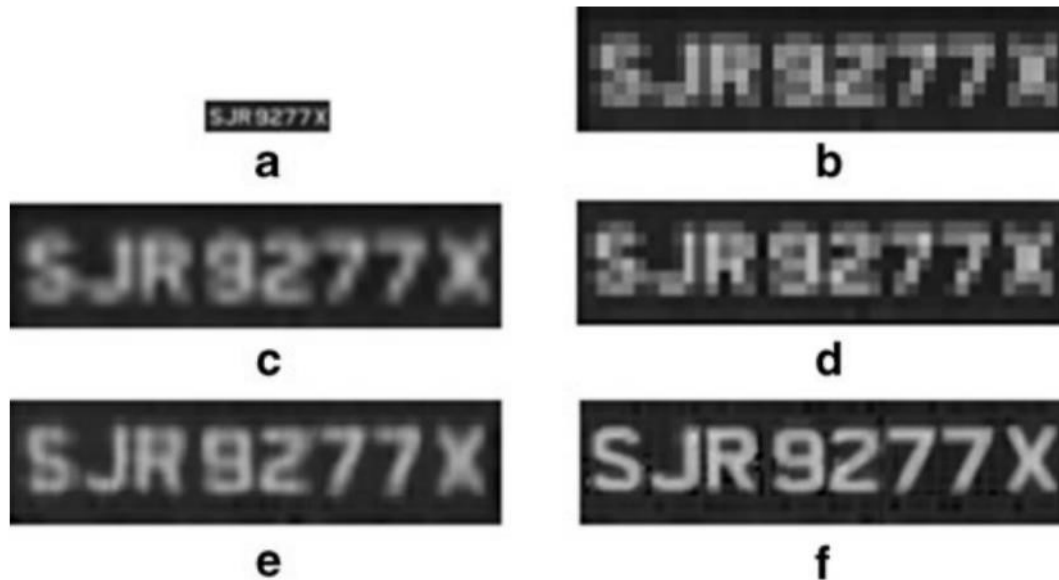


Figure 13: Sample Results of Method Proposed by Tian et al. (2012, p. 529), being image (f)

Using an OCR round during training appears to have been a fruitful methodology for their research. Figure 13 shows that their proposed method reduced the amount of ‘fuzziness’ around the edge of the characters. The use of an OCR stage in training would be analogous to using an OCR match as the discriminator of a SRGAN. In the case of this (the Optometer) project, ESRGAN is relativistic and would require some significant rewriting to include OCR as part of the discriminator stage. There would also be the question of how to relativise the OCR output, or whether a relativistic GAN would adapt in any effective way into such a process.

Table 31: Performance Comparison of Optometer to Similar Research

Measure	Optometer	Tian et al. (2012)
PSNR	14.223	19.837

Quantitative comparison of this project and the work of Tian et al. (2012) shows that the more sophisticated OCR-based process yields significantly higher PSNR scores than this project was able to achieve.

In summary, the research by Tian et al. (2012) applies a more original and sophisticated method than the one proposed in this project, but both demonstrate the applicability of neural networks in improving legibility by single image super-resolution of a vehicle registration plate.

9.0 Conclusion

The objective of the Optometer project was to test the hypothesis that when executing on an image of a real license plate, a SRGAN trained on synthetic license plates as opposed to generic images will produce a better-quality output. To test the hypothesis, the Optometer project generated 50000 standard Western Australian license plates, and used a deduplicated subset for training, validation, and evaluation of pre-trained SRGAN models. The project limited license plates to the specific subset so that the datasets came within time and computing resource limitations. To evaluate the hypothesis, the project changed variables such as iteration count, and measured performance by average PSNR and SSIM over a control group of 100 license plates.

Of the two measures, SSIM more closely mirrored subjective human assessment of model performance. PSNR, not considering the features of the resultant image, tended to show that any of the models including the pre-trained ones, produced worse results than non-machine learning techniques, which contradicted the subjective assessments of result quality. Overall SSIM seemed to be an effective measure of model performance.

Experiments showed that although there were some local maxima and minima within the variable ranges, the result trends strongly indicated the usefulness of conducting further experiments at the higher iteration counts that were beyond the resources of this project. At the default training rate, the dataset size did not have a significant effect on model performance across the range of iterations tested. Increasing the learning rate identified the training variable set producing greatest improvement, but the results suggested that this variable produced coarse results. It is possible that there was an overtraining threshold within the experimental range. It is not known whether this threshold was subject to other experimental constraints such as dataset size.

For further work, this project recommends the application of more resources to enable more iterations over the training data. This report also recommends reducing the quality of training, testing, and evaluation images, to determine the efficacy of the system for more realistic subjects. The latter recommendation is based on the positive results that this project obtained when adding blurring to the evaluation input images.



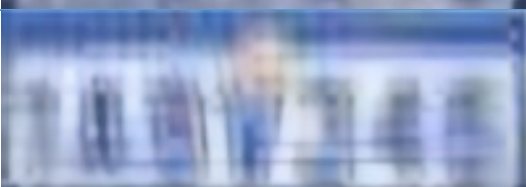


The SRGAN trained with synthetic data highly specific to the problem domain yielded subjectively and objectively better-quality outputs than the SRGAN trained on generic images. This tended to prove the hypothesis of the project.

10.0 References






- Capar, A., & Gokmen, M. (2006). *Concurrent segmentation and recognition with shape-driven fast marching methods*. Paper presented at the 18th International Conference on Pattern Recognition (ICPR'06).
- Dong, C., Loy, C. C., He, K., & Tang, X. (2015). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2), 295-307.
- Du, S., Ibrahim, M., Shehata, M., & Badawy, W. (2012). Automatic license plate recognition (ALPR): A state-of-the-art review. *IEEE Transactions on circuits and systems for video technology*, 23(2), 311-325.
- Duan, T. D., Du, T. H., Phuoc, T. V., & Hoang, N. V. (2005). *Building an automatic vehicle license plate recognition system*. Paper presented at the Proc. Int. Conf. Comput. Sci. RIVF.
- Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., . . . Bengio, Y. (2014). *Generative adversarial nets*. Paper presented at the Advances in neural information processing systems.
- Hore, A., & Ziou, D. (2010). *Image quality metrics: PSNR vs. SSIM*. Paper presented at the 2010 20th International Conference on Pattern Recognition.
- Jolicoeur-Martineau, A. (2018). The relativistic discriminator: a key element missing from standard GAN. *arXiv preprint arXiv:1807.00734*.
- Kanayama, K., Fujikawa, Y., Fujimoto, K., & Horino, M. (1991). *Development of vehicle-license number recognition system using real-time image processing and its application to travel-time measurement*. Paper presented at the [1991 Proceedings] 41st IEEE Vehicular Technology Conference.
- Lotufo, R., Morgan, A., & Johnson, A. (1990). *Automatic number-plate recognition*. Paper presented at the IEE Colloquium on Image Analysis for Transport Applications.
- Nomura, S., Yamanaka, K., Katai, O., Kawakami, H., & Shiose, T. (2005). A novel adaptive morphological approach for degraded character image segmentation. *Pattern Recognition*, 38(11), 1961-1975.
- Paliy, I., Turchenko, V., Koval, V., Sachenko, A., & Markowsky, G. (2004). *Approach to recognition of license plate numbers using neural networks*. Paper presented at the 2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541).
- Salomon, D., & Motta, G. (2010). *Handbook of data compression*: Springer Science & Business Media.
- Sarfraz, M. S., Shahzad, A., Elahi, M. A., Fraz, M., Zafar, I., & Edirisinghe, E. A. (2013). Real-time automatic license plate recognition for CCTV forensic applications. *Journal of real-time image processing*, 8(3), 285-295.
- Tian, Y., Yap, K.-H., & He, Y. (2012). Vehicle license plate super-resolution using soft learning prior. *Multimedia Tools and Applications*, 60(3), 519-535.

- Wang, X., Yu, K., Wu, S., Gu, J., Liu, Y., Dong, C., . . . Change Loy, C. (2018). *Esrgan: Enhanced super-resolution generative adversarial networks*. Paper presented at the Proceedings of the European Conference on Computer Vision (ECCV).
- Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4), 600-612.






11.0 Appendix 1: Sample Image of Iteration Tests over 1000 Training Images

Index	Model	Time	Mean PSNR (dB)	Mean SSIM	Sample
0	Resize	N/A	14.284954	0.207648	
0	RRDB_PSNR_x4.pth	N/A	14.231533	0.232456	
1	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 1000 Iterations	Not Measured	14.213339	0.246539	
2	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 2000 Iterations	Not Measured	14.180878	0.238300	
3	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 4000 Iterations	Not Measured	14.219912	0.239765	





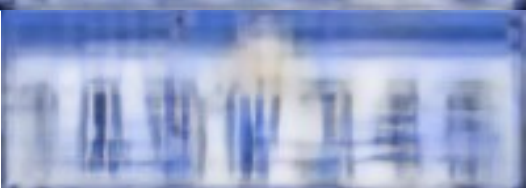
CSG3303 – Applied IT Project
Optometer -Analytical Project Report

Index	Model	Time	Mean PSNR (dB)	Mean SSIM	Sample
4	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 8000 Iterations	Not Measured	14.085603	0.248589	
5	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 16000 Iterations	178m44.281s	14.255639	0.254330	
6	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 32000 Iterations	364m36.864s	14.309129	0.256638	
7	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 64000 Iterations	709m47.099s	14.145575	0.251079	
10	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 128000 Iterations	1434m14.311s	14.273890	0.251808	

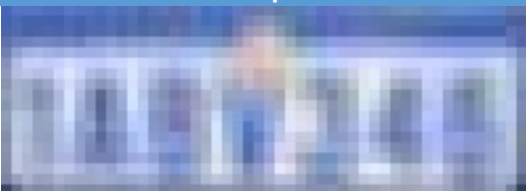




12.0 Appendix 2: Sample Image of Iteration Tests over 45178 Training Images

Index	Model	Time	Mean PSNR (dB)	Mean SSIM	Sample
0	Resize	N/A	14.284954	0.207648	
0	RRDB_PSNR_x4.pth	N/A	14.231533	0.232456	
8	RRDB_PSNR_x4.pth 45178 training and 4524 evaluation +1000 iterations	12m52.710s	14.202454	0.244970	
9	RRDB_PSNR_x4.pth 45178 training and 4524 evaluation +2000 iterations	23m26.459s	14.169065	0.249539	
11	RRDB_PSNR_x4.pth 45178 training and 4524 evaluation +4000 iterations	45m58.777s	14.228152	0.247098	


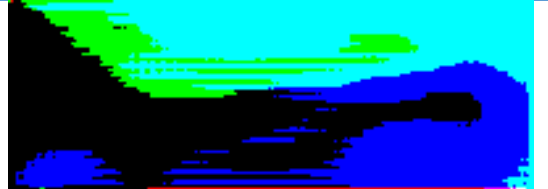
CSG3303 – Applied IT Project
Optometer -Analytical Project Report

Index	Model	Time	Mean PSNR (dB)	Mean SSIM	Sample
12	RRDB_PSNR_x4.pth 45178 training and 4524 evaluation +8000 iterations	98m16.589s	14.106782	0.252167	
13	RRDB_PSNR_x4.pth 45178 training and 4524 evaluation +16000 iterations	208m31.774s	14.183480	0.253738	
14	RRDB_PSNR_x4.pth 45178 training and 4524 evaluation +32000 iterations	411m46.666s	14.215105	0.261269	
15	RRDB_PSNR_x4.pth 45178 training and 4524 evaluation +64000 iterations	810m59.865s	14.287242	0.254009	
16	RRDB_PSNR_x4.pth 45178 training and 4524 evaluation +128000 iterations	1615m16.178s	14.242665	0.255075	






13.0 Appendix 3: Sample Images of Learning Rate Tests of 1000 Images over 1000 Iterations

Index	Model	Time	Mean PSNR (dB)	Mean SSIM	Sample
0	Resize	N/A	14.284954	0.207648	
0	RRDB_PSNR_x4.pth	N/A	14.231533	0.232456	
18	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 1000 Iterations + Generator Learning Rate = 1e-4 + Discriminator Learning Rate = 1e-4	11m21.572s	14.101473	0.246677	
19	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 1000 Iterations + Generator Learning Rate = 2e-4 + Discriminator Learning Rate = 2e-4	11m3.556s	14.061443	0.251790	
20	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 1000 Iterations + Generator Learning Rate = 4e-4 + Discriminator Learning Rate = 4e-4	11m28.423s	13.430871	0.256004	






CSG3303 – Applied IT Project
Optometer -Analytical Project Report

Index	Model	Time	Mean PSNR (dB)	Mean SSIM	Sample
21	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 1000 Iterations + Generator Learning Rate = 8e-4 + Discriminator Learning Rate = 8e-4	11m47.843s	5.837994	0.122854	
22	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 1000 Iterations + Generator Learning Rate = 16e-4 + Discriminator Learning Rate = 16e-4	11m21.132s	3.786857	0.037608	










14.0 Appendix 4: Sample Images of Iteration Tests for 1000 Images and 0.0004 Learning Rate

Index	Model	Time	Mean PSNR (dB)	Mean SSIM	Sample
0	Resize	N/A	14.284954	0.207648	
0	RRDB_PSNR_x4.pth	N/A	14.231533	0.232456	
20	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + 1000 Iterations + Generator Learning Rate = 4e-4 + Discriminator Learning Rate = 4e-4	11m28.423s	13.430871	0.256004	
23	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + Generator Learning Rate = 4e-4 + Discriminator Learning Rate = 4e-4 + 2000 Iterations	22m13.343s	14.130483	0.255801	
25	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + Generator Learning Rate = 4e-4 + Discriminator Learning Rate = 4e-4 + 4000 Iterations	43m24.701s	14.179063	0.247820	

CSG3303 – Applied IT Project
Optometer -Analytical Project Report

Index	Model	Time	Mean PSNR (dB)	Mean SSIM	Sample
26	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + Generator Learning Rate = 4e-4 + Discriminator Learning Rate = 4e-4 + 8000 Iterations	90m18.713s	14.327632	0.257626	
	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + Generator Learning Rate = 4e-4 + Discriminator Learning Rate = 4e-4 + 16000 Iterations	174m55.522s	13.706979	0.252617	
	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + Generator Learning Rate = 4e-4 + Discriminator Learning Rate = 4e-4 + 32000 Iterations	347m41.739s	14.220140	0.258228	
24	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + Generator Learning Rate = 4e-4 + Discriminator Learning Rate = 4e-4 + 64000 Iterations	706m25.368s	14.223097	0.280424	
	RRDB_PSNR_x4.pth 1000 training and 100 evaluation + Generator Learning Rate = 4e-4 + Discriminator Learning Rate = 4e-4 + 128000 Iterations	1403m33.708s	13.793866	0.234346	

15.0 Appendix 5: Quality Comparison as Input Image Degrades

Blur Layers	Sample Input	Mean PSNR (dB)	Mean SSIM	Sample Output
0		14.223097	0.280424	
1		15.566439	0.372767	
2		15.538547	0.354019	
3		15.666177	0.348821	
4		15.783697	0.340570	