A project submitted in partial fulfilment of the award of the degree of Bsc (Hons) Computer Games Programming From Staffordshire University

Supervised by Dr. David White

Assessed By Thomas Oliver

May 2021

A Self-Driving Neural Network to handle multiple off-road terrains

By David Taylor 18013908 BSC (Hons) Computer Games Programming

Table of Contents

[1 Abstract 4](#_Toc71639371)

[2 Introduction 4](#_Toc71639372)

[2.1 Aims and objectives 4](#_Toc71639373)

[2.2 Legal and Ethical Considerations 4](#_Toc71639374)

[2.3 Risk Assessment 5](#_Toc71639375)

[2.4 Project Management 5](#_Toc71639376)

[2.4.1 Considered Methods 5](#_Toc71639377)

[2.4.1.1 Agile 5](#_Toc71639378)

[2.4.1.2 Waterfall 5](#_Toc71639379)

[2.4.1.3 PRINCE2 5](#_Toc71639380)

[2.4.2 Chosen Method 5](#_Toc71639381)

[3 Context and Preliminary Investigation 5](#_Toc71639382)

[3.1 Problem definition and Rational 5](#_Toc71639383)

[3.2 Project background 6](#_Toc71639384)

[3.2.1 Similar projects 6](#_Toc71639385)

[3.2.2 Background Research 6](#_Toc71639386)

[3.3 Investigation method 8](#_Toc71639387)

[3.4 Project Planning 8](#_Toc71639388)

[3.4.1 Gant Chart 8](#_Toc71639389)

[3.4.2 Required Tools 9](#_Toc71639390)

[3.4.3 Deliverables 9](#_Toc71639391)

[3.4.4 Success Criteria 9](#_Toc71639392)

[4 Analysis 9](#_Toc71639393)

[4.1 Analysis method 9](#_Toc71639394)

[4.2 Narrative description of the analysis method 9](#_Toc71639395)

[5 Design 10](#_Toc71639396)

[5.1 Tools Research 10](#_Toc71639397)

[5.1.1 Engine 10](#_Toc71639398)

[5.1.1.1 Unity 10](#_Toc71639399)

[5.1.1.2 Unreal 10](#_Toc71639400)

[5.1.1.3 Chosen Engine 11](#_Toc71639401)

[5.1.2 API 11](#_Toc71639402)

[5.1.2.1 TensorFlow 11](#_Toc71639403)

[5.1.2.2 Open AI 11](#_Toc71639404)

[5.1.2.3 Chosen API 12](#_Toc71639405)

[6 Implementation 12](#_Toc71639406)

[6.1 Development plan 12](#_Toc71639407)

[6.2 Overall Structure And Background Code 12](#_Toc71639408)

[6.2.1 C++ 12](#_Toc71639409)

[6.2.2 Blueprints 13](#_Toc71639410)

[6.2.3 Python 13](#_Toc71639411)

[6.3 Neural Network Versions 14](#_Toc71639412)

[6.3.1 Version 1 14](#_Toc71639413)

[6.3.1.1 Features 14](#_Toc71639414)

[6.3.1.2 Usefulness 15](#_Toc71639415)

[6.3.2 Version 2 15](#_Toc71639416)

[6.3.2.1 Features 15](#_Toc71639417)

[6.3.2.2 Usefulness 16](#_Toc71639418)

[6.3.3 Version Three 16](#_Toc71639419)

[6.3.3.1 Features 16](#_Toc71639420)

[6.3.3.2 Usefulness 16](#_Toc71639421)

[6.4 Development Issues 16](#_Toc71639422)

[6.4.1 Hardware Issues 16](#_Toc71639423)

[6.4.2 Software Issues 16](#_Toc71639424)

[6.5 Data Sets 16](#_Toc71639425)

[6.5.1 Data set 1 16](#_Toc71639426)

[6.5.2 Data set 2 17](#_Toc71639427)

[6.5.3 Data set 3 17](#_Toc71639428)

[6.6 Car System 17](#_Toc71639429)

[6.6.1 Sensors 17](#_Toc71639430)

[6.6.2 Score 17](#_Toc71639431)

[7 Testing 17](#_Toc71639432)

[7.1 Testing Method 17](#_Toc71639433)

[7.2 Testing planning 18](#_Toc71639434)

[7.2.1 Forza Horizon 4 Testing Area: 18](#_Toc71639435)

[7.2.2 Software Artifact Testing Area: 18](#_Toc71639436)

[7.3 Conclusion From Results 18](#_Toc71639437)

[7.3.1 Forza AI performance 18](#_Toc71639438)

[7.3.1.1 Normal AI 18](#_Toc71639439)

[7.3.1.2 Drivatars 18](#_Toc71639440)

[7.3.2 Project AI performance 19](#_Toc71639441)

[7.3.3 Comparison between Forza AI and software artifact AI 19](#_Toc71639442)

[8 Critical Evaluation 19](#_Toc71639443)

[8.1 Success of the project 19](#_Toc71639444)

[8.1.1 Project Management 19](#_Toc71639445)

[8.1.2 Research 19](#_Toc71639446)

[8.1.3 Implementation 19](#_Toc71639447)

[8.1.4 Testing 20](#_Toc71639448)

[8.1.5 Overall project success 20](#_Toc71639449)

[8.2 Repeat of the project 20](#_Toc71639450)

[8.3 Extra Features 20](#_Toc71639451)

[8.4 Experience Gained 21](#_Toc71639452)

[8.4.1 Academic 21](#_Toc71639453)

[8.4.2 Professional expertise 21](#_Toc71639454)

[8.4.3 Programming 21](#_Toc71639455)

[9 Conclusion 22](#_Toc71639456)

[10 User Guide 22](#_Toc71639457)

[11 Bibliography 23](#_Toc71639458)

[12 Appendix 25](#_Toc71639459)

# Abstract

Within open world games like Forza Horizon 4 and GTA V the world is populated with self-driving car AI’s. This issue is that current methods can break emersion when an AI car act irrationally or get stuck in random situations off-road, this break in immersion can destroy player’s experience of a game. This study aims to develop a new method of car AI for open world games with off road terrains, specifically A neural network to handle off-road terrains. This paper discusses the implementation, tools used and a comparison of the AI directly to an area in Forza Horizon 4 to test if the AI can perform better or worse.

# Introduction

Over the last few year neural networks using Deep learning to train have achieved breakthroughs in difficult tasks. From ground-breaking “image classification” and “speech recognition” (Chollet, 2017) to “Superhuman Go playing” and “Near-human-level autonomous driving” (Chollet, 2017).

With advances within three key fields the possibility for widespread use and implementation of an advanced car AI using Neural networks in games has become a possibility. This is currently achievable due to current GPU hardware improvements, advances within real life self-driving cars and games being published using “sophisticated machine learning technology” (Kwalee, 2019) like “Draw it” which uses an image recognition neural network. With all three fields shown to work separately the ability to implement an advance self-driving AI neural network into games without reducing framerates to below 30FPS is finally possible.

## Aims and objectives

This report aims to cover the research and development of A Self-Driving Neural Network to handle multiple off-road terrains. This report covers all steps of the research process and is as followed:

1. Legal and ethical considerations,
2. Context and Preliminary research,
   1. Neural Networks
   2. Self Driving Cars
3. Analysis of problem-solving method and Analysis method
4. Design of the software artefact
   1. Engine
   2. Neural Network API
5. Implementation of the software artefact
   1. Car Systems
   2. Neural Network Versions
6. Testing of the software artefact
7. Critical Evaluation

## Legal and Ethical Considerations

The legal and ethical considerations are tiny, this is due to the research looking into new methods for AI cars within video games. This research doesn’t physically effect anyone directly or indirectly and the research doesn’t affect or rase any issues in regards to ethical, social or environmental issues.

This research doesn’t re-use and previously collected data in any stage, all data required to train the neural network was collected within the artifact.

## Risk Assessment

A risk assessment was carried out at the start of this project and two risks were evaluated; The first risk was the process of developing RSI from developing the software artifact. This risk can be prevented via the use of a ergonomic keyboard and mouse, This was unlikely to occour and is only minor severity. The second risk was eyestrain from researching and developing the software artifact. This is unlikely to occurs and is a minor in severity, this risk can be avoided by taking regular brakes whilst developing.

## Project Management

### Considered Methods

#### Agile

Agile project management main benefit is that its flexible, this allow changes to the project if large unexpected issues occur. The incremental iterative work process is beneficial allowing an easier time overcoming obstacle that could occur. Agile also allows the project to be altered if there is a necessary that wasn’t predicted early on in the project.

#### Waterfall

Waterfall project management is easy to understand and execute. All elements are extensively planned and requirements defined in full, This means there is almost no way to adapt to changes in the project or issues that arise.

#### PRINCE2

PRINCE2 project management method subdivides the project into smaller elements, each with independent plans and processes. This ensures the nothing in the project is left to chance as inputs and outputs for each section are predefined.

### Chosen Method

After reviewing each element, the best Project management method was Agile, mainly due to the ability to adjust current plans is vital as the implementation is not common and the best comparisons are for real life situations. This is not fully comparable to the limitations of running a similar simulation in video games. This means issues will occur during the project and agile is the best to adapt to those issues.

# Context and Preliminary Investigation

## Problem definition and Rational

The problem that this project is trying to solve is unrealistic AI cars in video games. This project is focusing on AI cars in off-road terrain and how they can react better to more situations. This type of AI can only recently been possible due to hardware developments in recent years. Improvement in car AI is currently not sort-after as most games focus on graphical improvements like raytraced reflections. This is the perfect time to do this type of research as the hardware requirements are becoming widespread with 16.25% use rate on the steam hardware survey (calculated on May 2021) (Steam, 2021). This value is only for computers running games through steam and doesn’t include the latest generation of game consoles. With the new “specialized hardware” (AMD, 2020) on the newest GPUs Nvidia with “Tensor Cores” (NVIDEA, 2020) and AMD with “Ray Accelerator” (AMD, 2020). These new compute style units can be leveraged for better AI performance. Currently using the compute style units would affect the compute units main focus of raytraced rendering and AI elements like “DLSS 2.0” (NVIDEA, 2020) to improve graphical fidelity. In the future it is highly likely for new GPUs to contain more compute style cores allowing this style of AI to become viable.

## Project background

### Similar projects

Currently the closest work is the software “Deepdrive” (Deeepdrive, 2020). This software is a realistic AI training software, It includes cars and uses unreal engine with python to develop the AI. This software does not include any AI and is only used to develop and train AI.

### Background Research

While researching for this project I used multiple papers talking and discussing topics related to and on self-driving cars. The first used paper for background research was Nvidia’s paper “End to End Learning for Self-Driving Cars” (Bojarski, et al., 2016). This paper covers the training structure of a “convolutional neural network (CNN) to map raw pixels from a single front-facing camera directly to steering commands”. This paper proved two important pieces of information, the first that they used a “simulation” to test the AI to ensure it worked, this proved that running AI in simulations to train could work. The second important piece is the papers reference to the “pioneering work” of Dean A. Pomerleau Autonomous Land Vehicle in a Neural Network (ALVINN) paper discussed below.

The paper from Dean A. Pomerleau called Autonomous Land Vehicle in a Neural Network (ALVINN) was critical to this project as it demonstrates that an end-to-end (An end-to-end neural network is where the neural network takes raw input and outputs the correct data straight away) neural network can control a car. This is a paper discussing a real-life implementation of a self-driving car. This AI model used an “30x32 Video Input Retina” and a “8x32 Range Finder Input Retina” (Pomerleau, 1989). This model was successfully able to drive “ at a speed of 1/2 meter per second along a 400-meter path through a wooded area” (Pomerleau, 1989). This report was used even due to its age as it was the best overall report. Due to its successful outcome and its coverage of all required components it was a perfect place to start my research.

The third paper used to research this area was a paper called “A Survey of Deep Learning Techniques for Autonomous Driving” (Sorin, et al., 2020). This paper discusses multiple methods for AI self-driving cars and their inputs. This paper covers 12 different self-driving car methods and what their aim is. This report also covers multiple data sets for different problems including their location, scenery, size and sensor setups.

The next main resource for research was “Deep Learning With Python” (Chollet, 2017). This is a book by François Chollet. The creator of the Keras API, commonly used for AI neural networks implementations. The first four chapters of this book cover all required knowledge like “High-level definitions of fundamental concepts” “Timeline of development of machine learning” “Key Factors behind deep learnings rising popularity and future potential” (Chollet, 2017, p. 3)

The key components that the book covers in chapter one and is critical for readers to understand the rest of the paper are:

* Difference between Artificial intelligence, Machine learning and deep learning.
* Machine learnings different “programming paradigm”
* The three components that construct deep learning

The key elements that separate Artificial intelligence Machine Learning and Deep Learning are that Machine learning is a sub-field of artificial intelligence and Deep learning is a sub-field of artificial intelligence. (Figure 1.0). This is critical to understand as there are multiple methods that can be called Artificial intelligence like behaviour trees.

The next key element to understand how the machine learning is different to classical programming. In classical programming data and rules are processed for answers(Figure 1.1). An example of this would be boiling a kettle, the data passed if would be the water temperature and if the kettles on or off. The rules would be to turn on the kettle if the waters cold, and to pour the water into a mug if its warm. Now how machine learning takes in the data and the answers these are processed to output a set of rules an AI can follow. Taking the example from before, you would pass in the temperature of the kettle and if its on or off, but you would then pass in when to turn the kettle on and when to pour the water into a cup. This would output the rules to turn on the kettle if the waters cold, and to pour the water into a mug if it is warm.

The third important element to understand is the three components of “deep learning” (Chollet, 2017, p. 8) are Weights, Loss function and an Optimiser. Each layer of the neural network contains weights, these weights slightly alter elements within the layer. The next key element is the loss function, this elements takes the correct output provided and compares to the AI’s output. This comparison is called a loss score. This loss score is then used by the optimizer to change the layers weighting to provide a better output. This process is tried over and over until the AI’s output is accurate enough.

The next key elements that the book covers in chapter two are:

* Tensors
* Tensor Operations
* How neural networks learn through backpropagation and gradient descent

The books second chapter covers “tensors that are 0D to 4D” and mentions the use of 5D tensors when processing video data. Tensors are simply groups of data, for example a 1D tensor is constructed of 0D tensor (singular piece of data). A method to understand the tensors dimensions is using paper, a singular sheet of paper can be written on left to right, up to down. This is like a 2D tensor, once you add another piece of paper behind the original one, you can write left to right, up to down and on a different page this would be a 3D tensor.

Each tensor contains three key elements, Number of axis, Shape, and data type. Following the previous analogy. The number axis would be how many options do you have to right, which would be three(left to right, top to bottom, what page).

The number of axis is how many dimensions the tensor has, the analogy above contains three dimensions. One dimension which is represented by a singular line of characters, two dimensions represented by multiple lines and three dimensions represented then multiple pages. The shape of the tensor is the number of elements per dimension this would the number of characters per line, the number of lines and the number of pages in the example. The last key element is the data type, this is what is stored in the tensor and would be represented by what you write on the paper.

The next elements discussed is high level view and explanation of tensor operations and like “Tensor dot”. This is the “most common, most useful operation” (Chollet, 2017, p. 40) (Figure 1.3).

The final discussed element in chapter two is how backpropagation and stochastic gradient decent can be used to improve a neural network. Backpropagation “sometimes called reverse-mode differentiation” (Chollet, 2017) is used to calculate what elements within the neural network are performing the worst. This is then used by Stochastic gradient decent or SDG, to alter the elements that are performing the worst.

SDG works through a derivative of a single coefficient can be interpreted as a slope of a curve ( 1.4). This curve can be used to optimise the Neural network via sampling the curve in steps in search for the lowest point. With SDG there is an issue called local minimum(Figure 1.5). This is where there is a small dip in the curve and then an increase and then a much larger dip after it. This issue can be solved with momentum. A good method to understand momentum is a ball rolling down the line, if the ball has enough momentum it can move up the slope to a new low point. This is repeated until the momentum of the ball cannot overtake the slopes increase.

The third chapter cover similar areas as chapter one and two just with real life examples using keras with python. These examples are more common implementations for AI like “Classifying movie reviews as positive or negative(binary classification)” “Classifying news wires by topic (multiclass classification)” “Estimating the price of a house, given real-estate data(regression)” (Chollet, 2017, p. 57)

The fourth chapter covers most importantly tackling overfitting. This is a critical element of neural networks as overfitting means that the AI can only work in its training scenario. So if the AI is trained to boil water in a kettle it could only boil water in that specific kettle and any other brand of kettle would result is errors like cold water or spilling water when using the kettle. This is unwanted behaviour so you must prevent overfitting.

## Investigation method

The investigation method used in this report is the researching methodology. This requires the four stages, focusing and planning, sourcing information, Analysis, Reporting.

This methodology has been followed throughout planning, An example of focussing and planning is decided what element I wanted to focus on as its not possible to create a physics engine, car steering and physics and AI all at once in the time provided. That’s why the project is focusing on the AI implementation using whiles using an game engine for the physics and car elements in addition with a machine learning API.

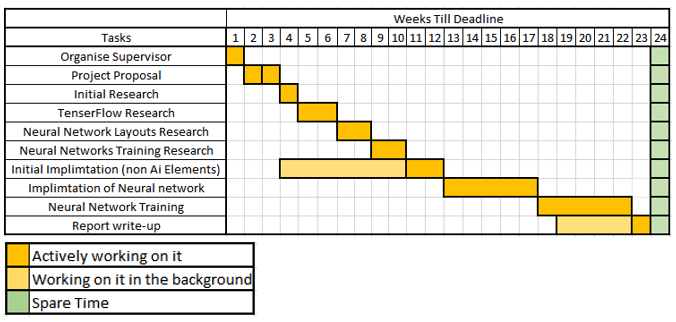
An example of sourcing information is all the background research into machine learning and deep learning and different methods discussed above in the background research section. This research is from multiple different publications and formats to show what ideas are accepted within the field.

The analyst stage within this project is represented as the planned design, this plan researches and discusses the benefits of all possible elements from what machine learning API to what game engine to use.

The last stage of the research methodology is the software artefact and this report.

## Project Planning

### Gant Chart



This Gant chart covers all the key tasks and the total amount of weeks to implement each element (number of weeks = university week). The creation of the Gant chart benefits project execution as all tasks and time allocated to the tasks are easily visualised.

### Required Tools

To complete this project six tools are required these are:

* Word Processor
* Computer
* Internet Access
* Game Engine that handles physics and car
* Machine learning API
* Visual studio 2019

These elements are the necessities for the project to progress and to be achievable.

### Deliverables

The items that will be delivered for submission are:

* All documents’ requirements stated in the Final Year Project Handbook
* Logbook
* All appropriate design and implementation documentation relevant to the selected methods
* A full program listing
* Installation and user documentation
* Original Project Plan
* Software Artifact (Unassisted movement, Safely traversing over altering terrain and movement to a predetermined location)

### Success Criteria

The success criteria are:

* Identification of a suitable method to use in the software artifact
* Producing a requirements specification that identifies the problem and justifies it
* A working prototype that addresses the problem and is within specification (Points mentioned above) and applying an appropriate design method.
* Undertaking appropriate testing strategies

# Analysis

## Analysis method

The analysis used while developing the software artifact was quantitative analysis via the AI score. The quantitative analysis method focuses on statistics measured accurately and fairly to uncover patters within the data.

## Narrative description of the analysis method

An example of quantitative analysis method being used is with the Automatic testing implemented in the first version of the AI. This method calculated a score for the AI based of distance travelled \* laps completed. This data is stored within a json file format for every version tested (Approx. 9,000 different version). This data uncovered a pattern of favourite attributes. (Figure 2.0)

(TODO: problem identified and overcome)

# Design

## Tools Research

### Engine

For this project to reduce scope an engine is required to handle the physics and car elements like steering and acceleration. For this section there will be an in-depth review of Unity and UE4, these are the engines reviewed as they are commonly used. This choice is key as it will critically effect the outcome of the project via its quality and useability.

#### Unity

The unity engine is a games engine with the ability to develop 2D and 3D games. The engine uses a C# Scripting language (Unity, 2020) to program game behaviours. This section will cover the positives and negatives of the engine for the specific application of A Self-Driving Neural Network to handle multiple off-road terrains.

##### Positive

The Unity engine has lots of positives, but this report will only cover useful features to neural networks. The first positive is the quality and amount of documentation, the documentation provided is easy to understand and can be quickly used to fix issues or to learn how to use a new system. This would improve debugging speed and implementation speed.

The next positive is Unity’s “Machine Learning Agents Toolkit ” (Unity, 2020), this is an addition to the engine that allows the use of machine learning to teach AI agents. This Toolkit comes with a few different implementations as well as the ability to use “TensorFlow” (Tensorflow, 2020) Python scripts to implement completely custom AI’s.

The last positive related to this project is Unity’s built-in physics engine. This removes reduces the scope of the project dramatically and thus making the project more likely to succeed.

##### Negative

With any elements of software there are benefits and negatives, the negatives of unity this time around are large. The largest issue is the lack of pre-made car behaviours, this increases the scope dramatically thus making the project less likely to succeed.

The next issue is that the “Machine Learning Agents Toolkit” is lacking in quality and useful documentation, this documentation would increase the difficulty in developing the AI and take extra time to fix any issues that arise.

#### Unreal

“Unreal Engine 4” (UE4) Is a 3D game engine that uses c++ and a visual scripting language called “Blueprints” (UE4, 2020). UE4’s source code is also freely available to alter and change the engine.

##### Positive

The positives of UE4 is that It comes packaged with lots of features ready to implement, an example of this is the Vehicle Advanced Starting project. This project contains an advanced physics-based car with suspension and steering already implemented. This ability removes the need to create any physics or car system removing scope to the desired level.

Another positive of UE4 is a plugin called “Tensorflow plugin for Unreal Engine 4” (getnamo, 2016), this plugin uses multiple other to fully implement a python element into the engine allowing for full use of Tensorflow and other python specific functions. This allows full python functionality whilst allowing the use of TensorFlow/Keras documentation to implement an AI design.

##### Negative

There are multiple negatives of UE4 one is the engines documentation, this is just not up to standard can hinder progress and development time. This increased time required to fix new segments of code.

The next negative of the use of UE4 is the forced use of blueprints to communicate between the python and c++. This complicates system from two elements of the AI car system to three, this is negative, and it increases the scope as a data formatting element must be created.

#### Chosen Engine

After comparing Unity and Unreal Engine 4 and discussing the benefits of each engine. After the comparison, the better engine choice for this scenario is Unreal Engine 4, This is due to the freedom of UE4 python allowing for lots of access and the vehicle advanced starting package. The starting advantage provided by the base project Vehicle Advanced reduces the project scope massively providing the best chance to full complete the project.

The freedom provided by Unreal Engine Python is key ask it allows the full python experience from a custom python command console to update, change and install new components. This prevents any issues or restraints and is just a fully-fledged python environment.

### API

There are only two main machine learning API’s to use Tensorflow (Tensorflow, 2020) and OpenAI (AI, 2015). These API’s handle the complex math required for neural network and machine learning implementation. The reason that an API is implemented into the project is again due to size. These API’ increase the software artifacts development speed.

#### TensorFlow

Tensorflow is an “open-source platform for machine learning” (Tensorflow, 2020), Tensorflow handles the majority of the complex math and just allows the use to specify different key features like layer amount and connections and is developed/maintained by Google.

##### Positives

There are multiple positives to Tensorflow the main one is the in depth documentation and guides. These provide an amazing understanding and allow for quick fixes to be found and implemented, Another positive is that Tensorflow is backed by google, this provides a good team to push Tensorflow and funding that cause Tensorflow to get better version after version.

Another benefit of Tensorflow is the constant evolution of there API, currently they are working on a C++ version of Tensorflow to be published in the future. Due to this there is a large community and support when issues are encountered.

##### Negatives

There are not that many major issues with Tensorflow, The only possible issue the difficult installation procedure for GPU enabled learning as its required to install multiple different elements and one element on the wrong version will break everything. This type of issue can be difficult to fix but are still do able with the amazing documentation.

#### Open AI

Open AI is a machine learning API based mainly around text, Open AI is created by a group invested by Microsoft. This provides a large support to the API project. There is one major issue with Open AI is that the project is currently invite only and requires you to join a weight list, this API leverages the power of the cloud to perform features like Customer service, text field generation and more.

#### Chosen API

The decided upon API is TensorFlow, this is due to it superb documentation another benefit is being backed by google ensuring updates will keep improving Tensorflow in the future. Tensorflow also allows for GPU acceleration this a critical element that is needed to implement this style of AI into games. The large community of developers creates third party tutorials and resources, and the official learning guide recommends learning material allows for new users to learn and develop there first AI with Tensorflow. This would reduce the time required for this project as to learn would be easier to follow and understand. With the new development of a Tensorflow C++ version, more engines can start to implement Tensorflow natively and would not be required to install a plugin to get Tensorflow working.

# Implementation

The implementations of the AI car can be split into thee sections, the C++ Car section that handles all aspects of the car, The Blueprint section that takes the output from the Car and formats it and sends it to Python. The Python section contains the AI code. These sections will be discussed separately below.

## Development plan

Due to the large goal of the project, the development was split up into sections to ensure each system was working before adding any other abilities these sections were:

* Steering and Acceleration
* Steep Incline and declines
* Icy terrain

Splitting the traits of the AI into sperate versions to develop ensured each feature was working before starting on the next. Once that feature was fully working a backup would be created and the next feature implemented into the AI.

## Overall Structure And Background Code

This section will cover a high-level structural overview of the application. This will cover the three separate programming methods of C++, UE4 Blueprints and Python. This application required multiple languages to be completed as Tensorflow requires Python to be used.

### C++

The C++ area of code controls:

* Car Instantiation
* Car Steering
* Car Physics
* Car Score
* Car Reset
* AI Output Formatting

This AI script was originally part of the Vehicle Advanced starting project. This was largely refactored apart from the setup of the Cars steering and physics. From the base car eight sensing raycasts were added these are the inputs of the AI. These raycast are places in the eight nautical directions (north, north east, east, south east, south, south west, west and north west).

Other additions to the car are a on screen text to show real time updates on the distances, AI score, Laps, and time Stationary. These values where added to allow me to always know how the AI was doing even after not seeing the full run.

For the steering of the car the proved steering mechanism has be augmented to use the AI input instead of what key has been pressed. This was a simple change as it just requires a variable swap.

The car scoring was challenging to implement as the AI score should not increase for going round the course backward. There are four colliders along the track once the AI triggers all four in order the lap score multiplier is then increased.

The C++ script also handles resetting the car once it gets a signal from the Python script. This is a critical element for the auto testing feature used to find the best AI. This system was simple to implement as the car position rotation and speed just had to be reset.

The final element located within the C++ element is the formatting of the output passed back from the Python Script. This takes the response from the AI and convers a 0 - 1 scale to a -1 to 1 scale. This is a critical area of the code it makes the AI compatible with the car steering.

### Blueprints

The blueprint element of the project is a necessary requirement as the plugin “tensorflow-ue4” (getnamo, 2016) uses blueprint nodes to pass data into the Python script. So, within the blueprint there are three main sections, Updating Car, Formatting input data to json to pass into Python script And the return call from Python script.

Within the First area of the Blueprint the raycasts used for the neural network input are updated, then the car speed and finally the UI elements. This area is calling functions defined within the C++ section to ensure all the car related code would be located within the C++ script.

The next part of the Blueprint has two variations one that runs the AI to test and one that allows a human to control the car and record data. This logic switch is based on a Recording Data Mode a Boolean set at the start of every frame. If the “Recording Data Mode” is false, the raycast inputs are formatted and send into the Python script to be processed by the AI. If “Recording Data Mode” is true, then the inputs are passed into the Python script to be saved into a json file for future use.

The next element to execute is the update function defined in the python script via the “tensorflow-ue4” (getnamo, 2016) plugin node “Send Json Input” targeted to the Tensorflow component.

After the script, the AI output is Returned, if “Recording Data Mode” is true then do not do anything with the AI outputs. If “Recording Data Mode” is false, then the last bit of the Blueprint handles resetting the C++ code and applying the returned AI output.

### Python

The last section of the code implementation is the python script. For this script to be executed you need to place the name of the file within the Tensorflow component within the car. Within the Python file you need to inherit from “TFPluginAPI”. From this you must override the functions “onSetup()”, “onJsonInput()” and “onBeginTraining()”. A fourth function “getApi()” must be included for full functionality to work. onSetup is equivalent to a constructor, onJsonInput is equivalent to Update function and the onBeginTraining is an “expected optional api” (getnamo, 2016) function. This is meant to be used to place AI training code within but is not necessary.

Apart from the mentioned elements above no changes in python are necessary for implementation as all standard Python features are useable. Some small additions like writing to UE4’s logs is possible in Python too.

## Neural Network Versions

Whilst in development three large changes occurred with the AI, for this reason before each large phase. The next section will discuss in detail what made each version unique and the issue the new version was meant to fix.

This section will be covering the three different separate versions have been created, Each version is a large change from the previous method to resolve and issue with the AI. Each section will discuss the new features of that versions, Reason for new version and the useful ness of that version.

### Version 1

The first version of the AI Is a simple Neural network structure where the model can easily be edited by hand to optimise the AI’s performance. This was the first attempt at implementing an AI and was successful in providing self-movement.

#### Features

With the first version of the AI most of the features are not unique but the required features for the AI to drive the car. These features are:

* Data Recording and saving to Json file
* Save and Loading Weights to file
* Creation of Neural Network Model
* Loading training and evaluation data
* Training the constructed Neural Network
* Formatting Input data received from the C++ car code
* Processing Input data
* Formatting Output data to send to the C++ car code

The ability to Record Data used in training and evaluating the AI as its critical to teach the Neural network to move. This system stores the distance from each of the eight nautical Raycasts to a Json this is done every frame, once a set number of frames have passed the entire json structure is saved to a json file. In depth reviews of each data set is in a separate section below.

To ensure useability and consistency implementing a saving and loading weighting was essential. Getting the weightings from the neural network is handled through Tensorflow. This protects any AI progress against computer crashes and glitches. This is key to allow for fair and equal testing when comparing AI scores with other AI scores.

The creation of the neural network model is key as it is creating the neural network that will be run, For the first version of the AI this is located with onSetup(). This section of code has to be changed based on the wanted structure.

The loading and formatting of training data and evaluation data loads data from a specified Data set. This data is loaded in with the training data being formatted first and then evaluation data being formatted second.

Training of the built neural network requires two actions. The first action is to compile the neural network, during this you need to specify the selected optimiser and loss function. Once the model is compiled you can train the model using the “fit()” function. The fit function takes the training data, number of epochs, batch size and validation data. Once the fit function is called the AI will train causing the unreal editor to freeze until training is complete.

The final three methods combined make up the update loop for the AI, the first important feature is converting the input data located within a json to a numpty array. This requires converting the string json into a float and storing in a numpty array equal to the neural network input size. The next element in the update loop is using “predict()” to run the neural network. To get the results of the AI you pass in the previously formatted input results into the predict function and the outputs of the AI are returned. The last element is formatting the output of the predict function and returning in the formatted functions.

#### Usefulness

This initial implementation is critical as most elements discussed have stayed the same though out all versions. An example of elements that stayed the same though out all versions are the Update loop and Save and loading weights features.

The usefulness of the AI based on performance was disappointing, as even when spending three days altering the changing all aspects of the AI neural network. Since the AI showed no improvement during this time a solution to allow for automatic testing needed to be found.

### Version 2

The second version of the AI implemented Automatic testing. This automatic testing was implemented as a method to efficiently test new versions of the AI 24/7.

#### Features

* All features of version one with auto testing multiple values

The main feature with the second version is Automatic testing to find the best AI layout, this solution alters five different elements. The altered factors of the AI are:

* Epochs
* Amount of Training Data
* Sensor Input Selection
* Optimiser Selection
* Model Layout Selection

Epochs within the auto tester are tested within a range from 30 epochs to 5 epochs. These values were selected as when testing the first version AI by hand no improvements were visible after these values. The main goal of the range was to limit the number of variations needed to test the AI to ensure it was achievable within a week of processing time.

The second variable changed by the auto tester is the amount of training data used. This again is a range starting from the Amount of connection within a neural network + 15 to Amount of connection within a neural network - 15. This dependence on the number of connections within a neural network Is to ensure a fair and equal testing to the different AI structures.

The third changeable value by the automatic tester is the Sensor Input Selection. This selection has three sensor arrangement, one with all eight sensors, one with the front 5 and one with only the front three (Figure 3.0). This feature is necessary for the auto testing system to ensure that the input type wasn’t holding the AI back.

The fourth changeable value was the optimiser. There are two optimisers that the auto tester checks “Adam” (Keras, n.d.) and “SGD” (Keras, n.d.). These both showed promising results when used with the first version of the AI. Due to that success, they are the optimisers used within the automatic testing, including different types of optimisers would have been beneficial but would increase the time dramatically to complete testing.  
The fifth and final change the automatic testing changes is the internal structure of the AI. There are three different AI structures that the auto testing software is testing. They are:

* A Single layer with 8 nodes
* A Single layer with 16 nodes
* Two layers with 16 then 8 nodes.

Again, as before theses structured showed promise while testing the first version of AI.

#### Usefulness

The usefulness of this version is massive as it allowed constant 24/7 testing. The outcome of the AI Car is still bad as it still cannot complete a lap, however this testing is still extremely helpful as it rules out that the AI components are bad. This is a simple conclusion to make as none of the 9000 ish versions tested.

### Version Three

For the third version of the AI was a small change towards the end of the project was made. The change made was implementing a genetic algorithm. The reason to implement a genetic algorithm when focusing on backpropagation to teach, was to test if the input formatting could be causing the issue. This would test if the input formatting is the issue if the AI produced similar behaviour to AI generated in the Auto testing phase.

#### Features

The only feature implemented in the third version is the Genetic algorithm, This is implementation is using cross and mutation to improve the performance of the AI.

#### Usefulness

This version is very useful as it showed the issue currently in the project, Given more time I could implement a fix into the project to allow for a better performing AI.

## Development Issues

During the development of the software artifact, multiple issues occurred due to hardware / software. These issues couldn’t be though off of planned in advance, these issues delayed the entire project by multiple days/weeks.

### Hardware Issues

The main hardware issues were that the computer running the AI training was overheating. This extra heat overheated and destroyed. This required update to the computer hardware like a new case to improve airflow to solve the overheating and a new power supply unity to replace the broken one.

This issue caused a loss of two weeks whilst waiting for parts to be shipped and time taken to install the new components. This severely messed with the project plan as the project scope required all the time necessary to implement all elements of the software artifact.

### Software Issues

There was one software issue that was caused by the installation of “tensorflow-ue4” (getnamo, 2016), this issue was due to the version of numpy being too high. This error was not commonly known and required multiple days of research before finding a fix. This again threw off the timing of the project losing multiple days of development time.

## Data Sets

There are currently three different data sets within the application and this section will be covering each dataset.

### Data set 1

The first data was recorded while using a keyboard for steering input, this type of input results in the steering always being at maximum (1 or -1). This is not ideal as the ai would only be able to steer at completely left or right and be at max acceleration or max break. Another issue with the data is that the data set only contains data where steering and acceleration is active. Both issues will teach the AI that it must constantly steer and constantly accelerate/break.

### Data set 2

The second data set fixes all the issue that occurred with the first data set. A joystick was used to record the data to prevent the steering issue. The use of the joystick should encourage smooth steering with the AI. The fix for the next issue was to record all data and not limit to when inputs are active.

### Data set 3

The final dataset is a random mix of data set one to dataset two, this mixed dataset is called “DataScrambled.json”. The logic behind combining the data set one and two, is that the first data set consists mainly of steering situations but no situations going straight. Data set two mainly consists of driving straight with a few examples of steering. Combining both data sets would allow for a better overall data set.

## Car System

This section will cover the two important car systems, the Sensors and Score systems. These systems are implemented with the use of C++ code.

### Sensors

The sensors are very important to the AI as its how it interprets the world, currently eight sensors are hardcoded, and sensors placed through the use of UE4 engine. Each sensor is simply a “uSphereComponet”. This is used with UE4 raycast for the position and rotation to calculate the distance to the wall.

Each sensor also has a scanning distance that is currently set to 10,000. This is the perfect length to allow measuring the distance even on long straights, but not too long to effect performance negatively.

### Score

The score is calculated into two segments distance travelled and lap multiplier. Each section will discuss their implementation as well as the reasoning.

The first key element of score is the distance travelled this allows the AI to be marked and reviewed independently. Currently distance is calculated by adding the cars current speed to a variable. This is not accurate as the current speed is in miles per hour.

The second key element is the lap multiplier, this is calculated via four checkpoints around the course. If the checkpoints are triggered in order the lap multiplier is increased. The lap multiplier is applied every frame when calculating distance.

The score is used within the automatic testing system and as a fitness calculation for the genetic algorithm implementation. This method of implementing score emphasises the ability to complete a lap and not to glitch out of the map and drive infinitely.

# Testing

## Testing Method

The method of testing the project will be a direct comparison to a loop within Forza Horizon 4 game world. The testing method is a direct comparison to Forza as it is a triple A game that has realistic physics, graphics and even seasons but unrealistic AI cars that look like there on a rail even when a car is passing them at above 100mph. Another reason for reviewing the software artifact to Forza Horizon 4 is due to it being a perfect game for this type of AI to be implemented into due to the ability to race on any terrain type.

## Testing planning

The testing method that is used is a comparison of performance around an area selected within “Forza Horizon 4” game world, and the test track within the software artifact. This testing method will allow easy comparison with triple A quality AI cars against the developed AI car.

Due to the progress of the software artifact the track implemented within the software artifact only contains similar turns and not any terrain features like water crossings and inclines and declines.

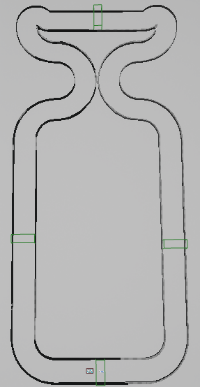
### Forza Horizon 4 Testing Area:

The route selected in Forza Horizon 4 is located on the outskirts of “LAKEHURST FOREST”. This area of the map is perfect for comparison as the loop contains:

* Water crossing
* Includes and declines
* Sharp turns
* Off-road Tracks (Dotted lines)

Within the local areas there are off-road steep terrain (Gray dotted line and red botted line). This would of allowed testing or more advanced terrain abilities if they were implemented. The combination of these traits makes a perfect testing area to compare all abilities that could be implemented into the AI.

### Software Artifact Testing Area:

The track within the software artifact contains the same types of turns found in the Forza Horizon 4 loop. This area was constructed to ensure as much matching steering traits with the provided track section within UE4. The shared qualities between the routes are:

* Sharp 90 degree turns.
* Large smooth turns
* Straights segments

## Conclusion From Results

Due to the testing method being

### Forza AI performance

#### Normal AI

The standard car AI in forza driving around the selected area takes its time all movement are deliberate following a predetermined set of actions. These are obvious to be on a predetermined track what the car models follow. The standard Car AI is emersion breaking due to there mathematical precision even when the environment is in chaos.

#### Drivatars

A custom type of car AI to Forza are the “Drivatars” (Team, 2019). A Drivatar is instantly different from any type of AI as they are more realistic and derived off recording of real players. Drivatars are multiple times better than the standard AI cars that populate the open world, this is mainly due to the slight imperfections of road positioning and variation from each Drivatars. The Drivatars increase the fun and players emersion into the game.

### Project AI performance

The performance of the software artifact is disappointing, the AI cant complete a full lap of the course, the AI has shown promise and progress multiple times during development but was held back by multiple issues like bad training and input data. If this AI was implemented into a driving game it would break emersion as it couldn’t progress down a simple lightly curved road well.

### Comparison between Forza AI and software artifact AI

While comparing both Forza AIs to the software artifact AI, a clear winner is shown. In the AI’s current state both Forza AIs are superior. This would quickly change once the software artifact AI is steering smoothly it would easily be a better experience more realistic AI than the Normal AI cars that populate the Forza game world. The Drivatars have one flaw that Forza needs to be connected to the internet to download the driver profiles for the Drivatars to work. At this point the fully working software artifact AI would beat Drivatars as the software artifact AI doesn’t require an internet connection to run.

# Critical Evaluation

## Success of the project

Each element of the project from project management to the testing will be reviewed and evaluated. Each section will be reviewed separately and then overall project review at the end.

### Project Management

The project management for this project was mainly excellent. This is due to use of agile method as it allows for adaptation of project when small errors occurred. This was critical as the implementation of the project was in completely using new API’s, Engines and Languages. The use of a Gant chart was very beneficial through out the project as it allowed easy viewing of upcoming tasks and the time frames to complete the task in.

The only issue with this project management method was not planning for any large unlikely issues like computers breaking. Due to this oversite two weeks were lost due to lack of equipment putting the project behind by two weeks early on.

### Research

The research conducted as part of this project was excellent, this is due to the wide range of research undertaken to ensure all sources were nonbiased. This was done by using multiple types of sources to ensure researching multiple authors papers. The best source for background research was “Deep learning with Python”. This was due to three elements, an amazing overall coverage of machine learning, excellent graphs and graphics showing a visual representation of the topic and the last real code examples. The only negative of my research should have been trying to find an paper on non-visual self-driving car implementations.

### Implementation

The implementation of the software artifact is bad when comparing against the success criteria directly. However, considering the unavoidable errors, lack of experience in the engine API and Python the implementation was successful. As even with this issue the final implementation was successful in a self-propelled AI car. The last major benefit for this implementation is the automatic testing for the car AI. This feature is set up to allow other types of optimisers, input layout, model structures to be changed.

### Testing

The method of testing the software artifact was a success as it compares the software solution directly to a modern driving game. This allows easy comparison as the tracks both AI are traveling down combines the same elements. Another benefit of using “Forza Horizon 4” for testing is that this is the type of game that would make the best use of the new type of AI proposed in this project. The issue with this testing method is that it is subjective to each individual this issue is attempted to be offset by ensuring both tracks have similar turns within them.

### Overall project success

The overall success of the project based on the success criteria is not the best as it only completes one of the criteria of unassisted movement. But when taking all errors that could not be controlled like computer overheating, PC part shortages delaying replacement parts and an Issue located within the install of “tensorflow-ue4” (getnamo, 2016). As well as considering little to no exposure in systems like UE4, Python and Tensorflow. What was developed was astonishing, the software solution has the ability to progress and even at this early of a stage shows how it could be a complete improvement for immersion with a more development time.

## Repeat of the project

If the project was to be reattempted three main elements would be altered, these are:

* Scope size
* Create a generic plan for large issues
* Use of the googles “Model Search” (Google, 2021)

Scope size would be reduced due to at first getting an AI car moving by itself, one this is working I can then update the project success criteria to match the time left. This change would allow the project to be more flexible and more likely to succeed.

The use of a genetic plan for large issues would be to combat the computer hardware failure that occurred during the project. Even if a loose plan were in place the issue could have been handled more effectively and quicker. This would reduce lost time due to the hardware failure.

The final and most interesting change would be to use and implement googles “Model Search” (Google, 2021) API. This would produce an excellent starting structure for the AI, from there the AI can be altered until the AI succeeds at its task. This would reduce the amount of time taken to produce and would have improved the project quality. The reason this wasn’t used in the project was that Model Search was released at the start of 2021, this was once the application was already in development.

## Extra Features

If the application was extended there would be three recommendations:

* Get the AI working on steep terrain and ice
* Implement support for more terrain surfaces like gravel and deep mud.
* Implement a method to support more devices

The first extensions to implement would be to get the car traversing over steep terrain and icy terrain. Once this is completed it would be a good idea to complete the test again, this would allow the developer to see if the AI method is better that once already implemented into forza.

The second extra feature is to implement more terrains like gravel and deep mud that the AI can drive over. This would increase the usability of the AI and thus increasing emersion. This would also show more of the possibilities of the AI and how good/bad it is.

The last extra feature is to implement a method to increase the number of systems that can run this AI. Currently there is a requirement of a Nvidia GPU with the compute drivers installed this can be a difficult and technical process. Due to this process being challenging some method needs to be created that automatically installs the required drivers when installing the game. Another method to increase the number of systems that support the AI is to get the AI running on AMD GPU’s. This extra feature would increase the usability of the AI massively as most modern gaming system could run it.

## Experience Gained

### Academic

The improvements to my academic knowledge gain while undertaking this project is massive, the areas that I have improved the most are:

* Legal and ethical considerations
* Time management
* Report writing

The first of the three areas that improved academically during this project is the legal and ethical considerations. For this project no direct or indirect risk to humans or animals, no environmental impacts and no use of previously collected data. However, learning about the ethical implementation of projects will be invaluable going forward. Ensuring all future personal research is kept to the same ethical standard as this report.

The second area that improved due to this project is Time Management. This has improved due to the experience gained via trying to time management a large project over one year. This experience has been invaluable as now future projects can be planned to remedy the issues present in this projects time management.

The third area that has improved is the ability to write a report, in both quality and speed. This is useful going forward in allowing faster documentation of new systems developed. The quality of writing over this project has slowly become better, this is critical as report writing has always been one of the worst areas.

### Professional expertise

This project improve the professional expertise on self-driving neural networks, this expertise can be used for multiple fields in the future from neural network development to in game Car AI development. This wide range of professional expertise was due to the development of this project.

### Programming

Whilst developing and implementing the project multiple new skills had to be learned to effectively develop the project. These newly developed skills are:

* UE4
* Tensorflow
* Python

The first area of improvement was in the use and understanding of UE4, this knowledge will be useful going forward as understanding a wide range of game engines allow quick adaptation to custom engines or codebases in the future.

The second area of improvement was going from knowing nothing about Tensorflow to being able to construct an AI that can run. Even If the final software artifact is not perfect the skills and knowledge of Tensorflow would improve future projects using the API.

The final improvement is going from no knowledge of python to being able to understand python. This is critical as understanding python is critical for the use and implementation of API’s like Pytorch and Tensorflow.

# Conclusion

This project has clearly shown that the implementation of A Self-Driving Neural Network to handle multiple off-road terrains is possible and has a high chance of success to complete with current AI implemented into triple A projects. If given more development time the project would flourish to fully handle multiple off-road terrains and be better or comparable to current AI car implementations. The extra functionality discussed would ensure that this method of AI works on all systems and greatly improve the AI off-road capabilities. These changes would make this method of car AI desirable for studios to develop.

# User Guide

# Bibliography

AI, O., 2015. *Open AI Homepage.* [Online]   
Available at: https://openai.com/  
[Accessed September 2020].

AMD, 2020. *RDNA 2 technologies.* [Online]   
Available at: https://www.amd.com/en/technologies/rdna-2  
[Accessed September 2020].

Chollet, F., 2017. *Deep Learning with Python.* 1st ed. s.l.:Mannig Publications.

Deeepdrive, 2020. *Deepdrive.* [Online]   
[Accessed September 2020].

getnamo, 2016. *Tensorflow plugin for Unreal Engine.* [Online]   
Available at: https://github.com/getnamo/tensorflow-ue4  
[Accessed September 2020].

Keras, n.d. *Adam Optimiser.* [Online]   
Available at: https://keras.io/api/optimizers/adam/  
[Accessed September 2020].

Keras, n.d. *SGD Keras Documentation.* [Online]   
[Accessed September 2020].

NVIDEA, 2020. *GEFORCE RTX 30 SERIES.* [Online]   
Available at: https://www.nvidia.com/en-gb/geforce/graphics-cards/30-series/  
[Accessed September 2020].

Pomerleau, D. A., 1989. *ALVINN: AN AUTONOMOUS LAND VEHICLE IN A NEURAL NETWORK.* [Online]   
Available at: https://apps.dtic.mil/sti/pdfs/ADA218975.pdf  
[Accessed Spetember 2020].

Sorin, G., Bogdan, T., Tiberiu, C. & Gigel, M., 2020. *A Survey of Deep Learning Techniques for Autonomous Driving.* [Online]   
Available at: https://arxiv.org/pdf/1910.07738.pdf  
[Accessed September 2020].

Steam, 2021. *Steam Hardware Survey.* [Online]   
Available at: https://store.steampowered.com/hwsurvey/Steam-Hardware-Software-Survey-Welcome-to-Steam  
[Accessed May 2021].

Tensorflow, 2020. *Tensforlow Homepage.* [Online]   
Available at: https://www.tensorflow.org/  
[Accessed September 2020].

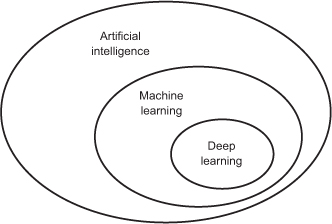
UE4, 2020. *UE4 Features.* [Online]   
Available at: https://www.unrealengine.com/en-US/features  
[Accessed September 2020].

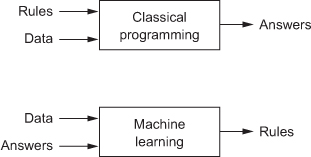
Unity, 2020. *Unity ml-Agents.* [Online]   
Available at: https://github.com/Unity-Technologies/ml-agents  
[Accessed September 2020].

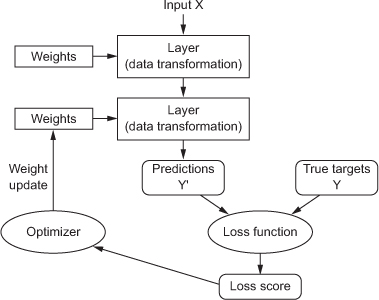
Unity, 2020. *Unity Platforrm.* [Online]   
Available at: https://unity.com/products/unity-platform  
[Accessed September 2020].

# Appendix

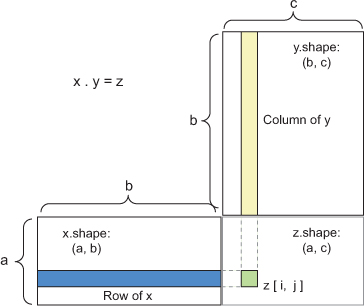
[Figure 1: The relationship between Artificial intelligence, machine learning and deep learning.

From Deep learning with python (Chollet, 2017, p. 4)]

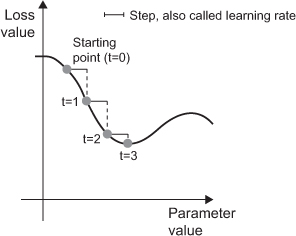
[Figure 1.1: The differences between Classical programming paradigm and Machine Learning paradigm (Chollet, 2017, p. 5)]

[Figure 1.2: Flow chart of deep learning (Chollet, 2017, p. 11)]

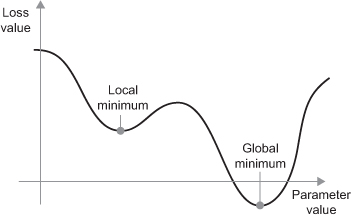
[Figure 1.3: Matrix dot-product box diagram (Chollet, 2017, p. 42)]



[Figure 1.4: SGD down a 1D loss curve (Chollet, 2017, p. 49)]



[Figure 1.5: Local and Global minimum (Chollet, 2017, p. 51)]



[Figure 2.0: Graph showing the top ten AI iterations and there traits]