Teaching computers to revel in glorious combat

A Capstone Project
by
Rei Armenia and Matthew James Harrison

Submitted to the Faculty of the
Department of Software Technology
in partial fulfillment of the requirements for the degree of

BACHELOR OF SCIENCE
IN
COMPUTER SCIENCE And INNOVATION

Champlain College
2017

Abstract

A brief description (sentence or two) of the problem and why the reader should care. A brief description of what you did. A purpose statement. Include clarifying context as necessary. A brief overview of the approach/method. A brief overview of the results.

Acknowledgements

This section is optional.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction (2-3 pages)

1.1 Problem Statement

Can we create a fair and balanced artificial intelligence that plays video games almost like a human?

1.2 Purpose Statement

We shall create an artificial intelligence that uses the same information available to players in order to make decisions. We shall also closely train the Agent to reproduce human-like gameplay on a reliable scale.

1.3 Context

One or two pages that introduce your project by providing important context. This section can include but is not limited to historical background leading up to your project, terminology, brief literature overview (save the details for the LITERATURE section), questions to be addressed, and basic capstone planning and method information.

1.4 Significance of Project

A paragraph or two that explains the significance of this capstone project. Why is this worth doing? What do you hope to gain? What do you hope to provide to the computing community?

Chapter 2: Literature (3-5 pages)

2.1 Literature Overview

In this section we will discuss and justify our project's software stack before briefly reviewing related research projects.

Here is an example citation [1].

2.2 Environment

We have opted to use our own personal computers running 64-bit Windows 10 with Ubuntu subsystems as both our platform and development environment. There are several reasons to justify this decision. First, we wanted to ensure that we would have the best possible experience when dealing with GPU computation, and NVIDIA simply has a longer and better track record for supporting Windows than it does for Linux. Second, we wanted to avoid the potential pitfall of our platform restricting us to a smaller library of target games. Our two top candidates were *Quake* and *Rivals of Aether* because both of these games support replays, or demos; however, only one of them natively supports Linux. Last, and with regards to using our own computers for both development and testing, we made the decision to abstain from distributed or cloud computing chiefly to save costs, but also as an aesthetic choice based in the desire to see what our computers are capable of.

Despite all of the above justifications for using Windows, it remains to be said that our project cannot live without Linux. For reasons discussed in the next section, we use a Redis server which runs on an Ubuntu subsystem. The solution may not sound elegant, but it was certainly simple to set up.

2.3 Software

Our project uses a software stack comprised of the following: SerpentAI, Keras, TensorFlow-GPU, and the Anaconda distribution for Python 3. We chose to use Python 3 as our primary programming language for its flexibility, intuitive syntax, and data processing capabilities. Python is also helpfully compatible with all of the other software listed above.

SerpentAI serves as a bridge between our program and its testing environment, providing a simple interface for retrieving frame buffer data from any given videogame runtime and then sending input data back. The Anaconda distribution provides us with a large assortment of Python modules, some of which SerpentAI cites as dependencies. This

allows us to more easily work from within a Windows environment, which in turn gives us the most straightforward access to GPU computation on our NVIDIA video cards, by way of NVIDIA's CUDA and CUDNN libraries.

2.4 Other Sources

Non-academic sources such as white papers, manuals, blogs, etc.

Chapter 3: Methods (5-7 pages)

3.1 Design

Design details of your project. This section can include various descriptions, figures, diagrams, wireframes, etc that are driving your project implementation.

3.2 Frameworks

Statement and descriptions of systems, languages, frameworks, APIs, and tools used in the development of your project.

3.3 Algorithms

Descriptions and pseudocode of important algorithms in use, relevant, or to be implemented as part of your project.

3.4 Analytical Methods

Descriptions and processes of any analytical methods used as part of your capstone project. This section is most applicable for projects with a data analysis focus or component. Quantitative and qualitative methods.

3.5 Features

A brief overview of primary features you plan to implement.

3.6 Test Plan

A brief overview of what and how you plan to test your project codebase.

3.7 Criteria and Constraints

Project development must meet desired needs within realistic constraints. Many times, constraints are interrelated. Cover the appropriate criteria and constraints related to your capstone. Some example criteria and constraints are health and safety, environmental, political, social, manufacturability, sustainability, economical, and ethical.

Chapter 4: Results (5-7 pages)

4.1 User Interface Design

Design details of your project. This section can include various visualizations (e.g., user interfaces, screenshots) that depict your final project implementation.

4.2 Architecture

Statement and descriptions of systems and structures (e.g., data structures) used in the implementation of your project. This section should detail the inter- and intra-workings of your software and the systems on which the software relies.

4.3 Algorithms

Implementations, explanations, and performance of important algorithms in use as part of your project.

4.4 Analytical Results

Results of quantitative and/or qualitative analysis. This section is most applicable for projects with a data analysis focus or component.

4.5 Features

Describe the planned features which were implemented. Briefly discuss the planned features that were not implemented.

4.6 Testing Results

A brief overview of what and how you tested your project codebase.

Chapter 5: Conclusion (2-3 pages)

5.1 Context

Provide historical, social, cultural, ethical, legal, and other relevant contexts for your project in a reflective manner.

5.2 Challenges and Solutions

Describe particularly important challenges encountered and solutions throughout the development of this capstone project.

5.3 Limitations and Delimitations

Limitations are aspects of a research study that the researcher cannot control. Delimitations are aspects of a research study that the researcher can control.

5.4 Future Work

Where would you like to take the project in the future?

5.5 Project Importance

Why does your project matter? What made this project an important investment of time and resources? Do not reiterate the significance from the Introduction. Rather, be reflective when discussing the project's importance. Discuss what you learned and what your project provides for the computing community.

References

[1] J. Doe, "An amazingly awesome paper," *IEEE Transactions on Awesomeness*, vol. 13, no. 2, pp. 321–332, 2017.

Appendix A: Additional Information

As necessary.