
Notes

- 10:20 AM
 - Iraq, A32, Atrium Side
 - 15 Minutes
 - w/ 5 Minutes Q&A
 - 10% Quality of work
 - 5% Quality of presentation
-

Cross-Game Generalisation Approaches for General Video Game Playing using Deep Reinforcement Learning

Benjamin Charlton
psybc3@nottingham.ac.uk

14262648

Computer Science with Artificial Intelligence MSci

Motivation & Background

Personal Interests

Games and AI

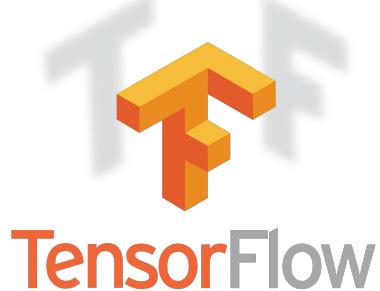
- Playing games
 - GameSoc
- AI
 - On the AI path
- Dissertation Project
 - Explored this area
 - Seen interesting ideas

GVGAI Competition

“explores the problem of creating controllers for general video game playing”



Familiarity w/ Machine Learning



- G53MLE
 - Brief intro to TF
- Complex topic
 - Warranted a deeper dive

Background

Game Playing and AI

- Board Games
 - Tic Tac Toe, Chess, Go
 - General Game Playing Competition
- Video Games
 - Atari, DotA 2, Starcraft II
 - GVGP

The Problem

GVGAI Tracks

- Single Player Planning
 - 2-Player Planning
 - Single Player Learning
 - Level Generation
 - Rule Generation
-

Game Playing Tracks

Same games different rules

- Planning
 - Current Game State
 - Forward Model
- Learning
 - Current Screen Image

Single Player Learning Track

Focuses on RL

- 3 Games
 - Levels 1, 2 - Training
 - Levels 3, 4, 5 - Validation
- OpenAI Gym Integration
 - Recent addition June 2018
 - Common RL framework

Approach

And some more reading

Reading



arXiv: 1806.02481 [cs.LG] 6 Jun 2018

I. INTRODUCTION

The influence of video game on game research and artificial intelligence research have even spread to the whole AI community, in particular since Chess and Go programs were solved by computer.

The following sections introduce the history of video game AI and the current state of video game AI research.

II. BACKGROUND

A. General Video Game AI

A general video game AI (GVGA) framework is intended to handle benchmarks for General Video Game Playing (GVGP) in 2-dimensional games and 3D games. This framework offers a 3-dimensional space for games. The framework offers two main ways to play games: one is to convert them to the Video Game Description Language (VGDL), which was developed by the authors of this paper, and the other is to convert them to the ALE (Automatic Level Editor) format, which was developed by DeepMind's Andrew Ng and others.

B. ALE

A General Video Game AI

As an AI benchmark, ALE is limited in the sense that there is only a finite set of games. This is a limitation if we want to evaluate the performance of an AI agent on a specific game. However, for being able to use the general video game playing framework, we need to convert the game to VGDL, which the agent will interpret. So far, we need to implement the conversion manually for each game. This is time-consuming and error-prone. We also follow the creating of games made by the same people who created the ALE games at the same time.

C. Deep Reinforcement Learning

Deep reinforcement learning (DRL) is a type of machine learning that allows agents to learn from their own experience. It has been used to solve complex tasks such as playing chess, checkers, and shogi. In recent years, DRL has also been applied to video games. The first breakthrough was the creation of AlphaGo, which defeated the world champion of Go. Since then, many other AI agents have been created, such as AlphaZero, which can play chess, shogi, and Go at a superhuman level. These agents are based on deep neural networks and reinforcement learning algorithms. The framework offers a 3-dimensional space for games: one is to convert them to VGDL, and the other is to convert them to ALE. This framework also follows the creating of games made by the same people who created the ALE games at the same time.

D. Related Work

There are several related works in the field of video game AI.

E. Contribution

In Table 1 we compare the performance of the as well

DRL shown to work ...

But only on individual levels and
games

- Torrado et al. shown the GVGAI Gym framework works
 - Brand new framework
- Didn't show generalisation
 - Levels
 - Games
- Solid starting point

Replicate Results

On a single game and a single level

- Used A2C Advantage Actor Critic
 - Multiprocessing
- Worked as expected
 - Some successes
 - Some failures

Issues

Hurdles to allow cross game play

- Technical Issues
 - Fixed Input Size
 - Fixed Output Size
- Data Processing
 - Scaling Rewards
 - Removal of alpha channel

Fixed Size

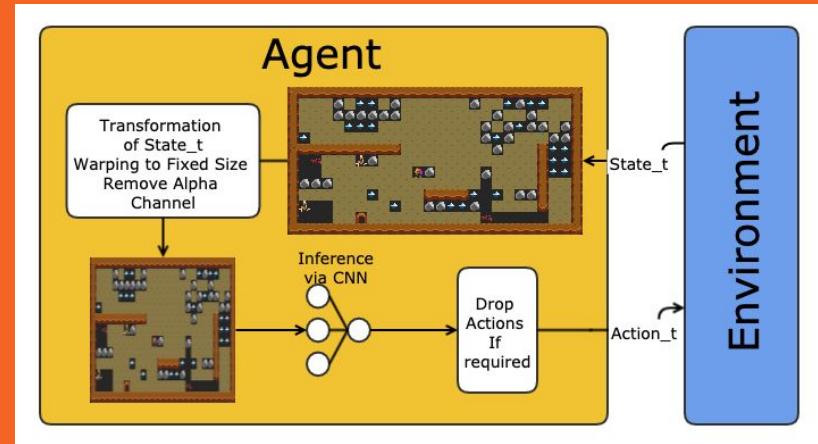
- CNN technically works on any size
 - Multi-Layer Perceptron requires fixed sizes
 - Process input to be predetermined size
 - Ignore unacceptable actions from the output layer
-

Data Processing

Increasing the efficiency of learning

- Scaling rewards
 - Normalisation
- Removal of alpha channel
 - Dimensionality reduction
 - Performed after data analysis

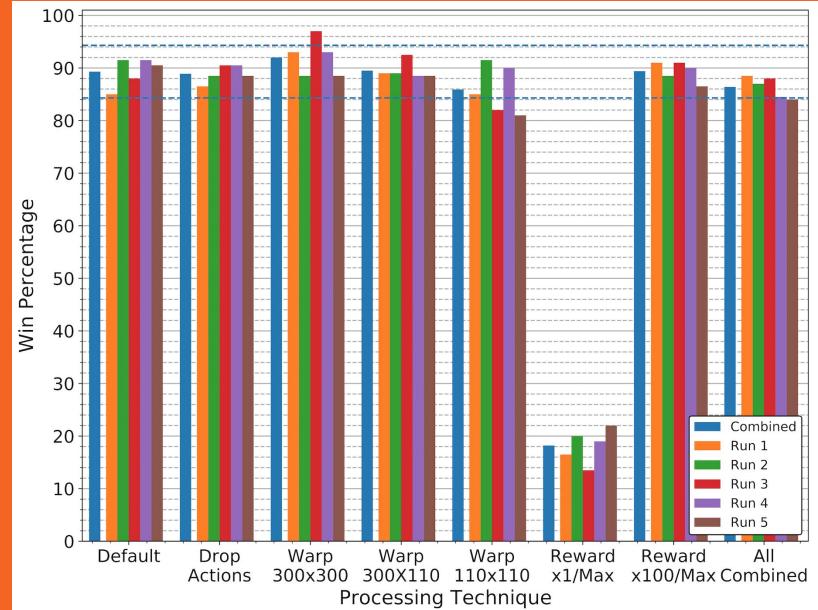
Agent Structure



Results and Conclusions

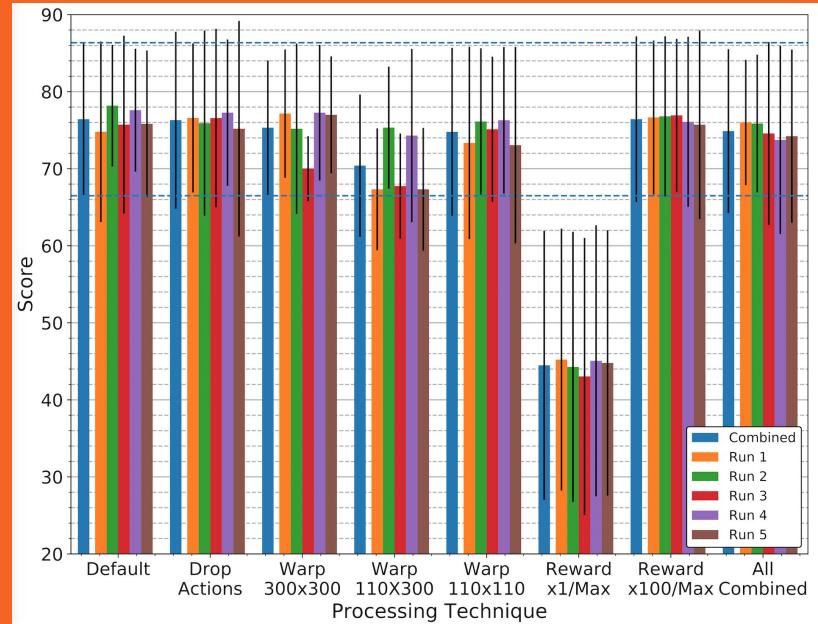
Processing Results

Win%



Processing Results

Score



GVGP

Performance

Qualitative Results Aliens

- Strong on training and validation
- Shots not 100% accurate
- Performs worse when aliens lower
 - Levels spawn from upper left
- Destroys barriers

GVGP

Performance

Qualitative Results Missile Command

- Weak on levels w/ blue missiles
 - Not in the training set
 - Simply ignores them
- Strong on final level
 - Very similar to first level
 - Increased difficult not an issue

GVGP Performance

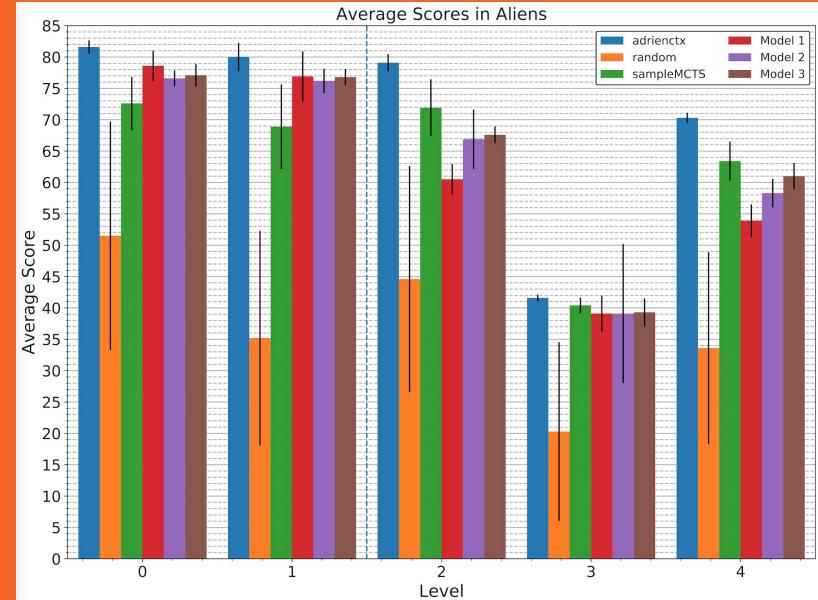
Qualitative Results Boulderdash

- Gained high scores in training
- Struggled to win on any level
 - Win condition not tied to reward
- Appears to overfit training
 - Few random elements

GVGP

Performance

Quantitative Results



Conclusions

- NN can work with minor modifications
 - No forward model required
 - Screen Observation is sufficient
 - Fixed Inference time
 - Not Explainable
 - Expensive Computational Requirements
-

Future Work

- Improvements to training methods
 - More sophisticated NN
 - Feed in temporal info
 - RNNs
 - Transfer Learning
 - Online Learning
 - Procedural Content Generation
 - Data augmentation
 - Submitted to IEEE CEC Competition
 - IEEE COG
-

Thank You

Q&A