Balloma Video Game playing Reinforcement Learning Agent.

Luis Rojas Aguilera
Udacity
Capstone Project Proposal

CONTENTS

I Introduction 1

II Deep Reinforcement Learning 1

III Balloma video game 2

IV Environment 2

V Reward function 2
V-A Your first solution . . . 3
V-B Your second solution . . . 3
V-C Your third solution . . . 3
V-C1 Subsubsection Heading Here . . . 3

VI Criteria for Assessing Solutions 3

VII Research Methodology 3

VIII Analysis and Interpretation 3

IX Conclusions and Recommendations 3

Appendix A: What Goes in the Appendices 4

Appendix B: Formatting the Appendices 4

References 4

LIST OF FIGURES

1 Balloma splash screen. . . . 1
2 Balloma's scene sample. Elements of scene are: a) The ball (blue), b) Floating elements (green), c) Score records (red), d) Target position (yellow) . . . 2

LIST OF TABLES

I Some impressive numbers . . . 3

Balloma Video Game playing Reinforcement Learning Agent.

***Abstract*—This work presents a capstone project proposal to achieve a Nanodegree in Machine Learning from Udacity. The objective of this work is to construct a video game playing robot, through the use of Deep Reinforcement Learning techniques. Also it should provide a methodology for test automation on game development process. This document provide descriptions on: *a)* problem to be solved, *b)* objectives, *c)* context, *d)* tools, *e)* metrics and *f)* techniques.**

I. Introduction

Balloma is a single-player, android video game, developed by Black River Studios[1] at brasilian SIDIA[2]. During the game development process, after any new features are integrated, regression and functional tests must be executed in order to validate if new functionalities are properly working and side effects caused by new code have not appeared.

Fig. 1. Balloma splash screen.

Currently such tests are executed manually by humans, thus as functionalities' stack increases also does testing complexity and workload. In order to aim testing process this proposal presents a Proof Of Concept method for test automation using Deep Reinforcement Learning techniques. Next sections presents further details on Balloma video game, the RL framework and how to use it for game automatic controlling.

II. Deep Reinforcement Learning

Reinforcement Learning (RL) is a field of Machine Learning that studies autonomous interactions among software agents and environments, and the way an agent can enhance its behavioral rules (policy) in order to maximize the reward it obtains from an environment. That is, at time step t, agent takes action a_t on a given environment E with transition dynamics $p(s_{t+1}|s_t, a_t)$ and current state s_t obtaining a reward r_t from a reward function $r(s_t, a_t)$ and transforming current state into s_{t+1}.

Currently RL proposes two main approaches to represent an agent behavioral rules: a) Value-based and b) Policy-based algorithms.

In value-based algorithms it is used a function called action-value ($q_*(s, a)$) that contains the expected cumulative reward of performing a given action while in a given environment's state. Such function can be used by the agent to decide which action to take in a given state by selecting from q the action that maximizes reward with a given probability ϵ that allows a desired exploration behavior so as it is applicable for non-deterministic environment dynamics. This approach is constrained to discrete action and state spaces.

Policy-based algorithms directly maps states and actions by a approximated policy function that represents the underlying stochastic distribution presented by environment's dynamics. While in the Value-based approach an heuristic should be defined to provide exploration behavior and avoid conditional randomness influenced by agent greediness, in Policy-based such heuristic is expressed in the policy function and is not necessarily fixed for each possible action-space setup. This approach is applicable to continuous action and state spaces.

For both approaches, agent behavior is guided by an optimal policy $\pi_*(s, a, \theta)$ evaluated by objective function $J(\theta) = \mathbb{E}[R(\tau)]$ of policy parameters θ, based on state-action-reward expectations on trajectory $\tau = S_0, A_0, R_1, S_1, ...$ drawn from a probability distribution $p(s', r|s, a)$. In such case is convenient using the **Bellman Expectation Equation** [5] to represent expected return of taking an action a_t while in state s_t and following a policy π:

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E}[r(s_t, a_t) + \gamma \mathbb{E}_{a_{t+1} \sim \pi}[Q^{\pi}(s_{t+1}, a_{t+1})]] \quad (1)$$

In this proposal I intend to apply a hybrid approach for RL called Deep Deterministic Policy Gradients (DDPG) [1]. It follows the actor-critic [2] algorithm DPG [3] in which the agent's behavior is represented by function $\mu(s|\theta^{\mu})$, called the actor, that is evaluated by the critic, a non-linear action-value function $Q(s, a)$ parametrized by θ^Q, adjusted by minimizing the loss:

$$L(\theta^Q) = \mathbb{E}_{s_t \sim \rho^{\beta}, a_t \sim, r_t \sim E}[(Q(s_t, a_t|\theta^Q) - y_t)^2] \quad (2)$$

where

$$y_t = r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1})|\theta^Q) \quad (3)$$

[1] http://blackriverstudios.net/
[2] https://www.sidia.com

Actor function $\mu(s|\theta^\mu)$ specifies the agent's policy by deterministically mapping states to a specific action. It is adjusted to maximize the expected reward from a start distribution $J = \mathbb{E}_{r_i,s_i\sim E,a_i\sim\pi}[R_1]$. That is attained by updating parameters θ^μ following the policy's performance gradients, expressed as:

$$\begin{aligned}\nabla_{\theta^\mu} &\approx \mathbb{E}_{s_t\sim\rho^\beta}[\nabla_{\theta^\mu}Q(s,a|\theta^Q)|_{s=s_t,a=\mu(s_t|\theta^\mu)}] \\ &= \mathbb{E}_{s_t\sim\rho^\beta}[\nabla_a Q(s,a|\theta^Q)|_{s=s_t,a=\mu(s_t)}\nabla_{\theta_\mu}\mu(s|\theta^\mu)|_{s=s_t}]\end{aligned} \tag{4}$$

Both actor and critic functions are expressed by neuronal networks, updated following an approximation to a supervised learning approach. For that matter, target networks $\mu'(s|\theta^{u'})$ and $Q'(s,a|\theta^{Q'})$ are created, which are soft copies of actor and critic networks respectively. That soft copy is made from updated parameters as follows: $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$ with $\tau \ll 1$

In order to provide exploration capabilities, during training, a different policy than that of the agent is used, obtained by adding noise sampled from a noise process η to the actor function. Such exploration policy is expressed as follows:

$$\mu'(s_t) = \mu(s_t|\theta_t^\mu) + \eta \tag{5}$$

Training follows an episodic process similar to Q-Learning. In each episode a set of agent-environment interactions, represented as a transition tuple (s_t, a_t, r_t, s_{t+1}) are stored in a replay buffer R [4]. A set of tuples are sampled from R and used to apply the updates previously explained in this section. This iterative process can be truncated based on optimization or time constraining conditions are met.

III. Balloma video game

Balloma is a single-player video game that runs on Android devices. It is composed of several scenes that initially appears locked and gets unlocked as the user completes unlocked scene's objective.

Each scene contains a constrained little terrain world with elements arranged and boundaries that if crossed makes the scene end. The scene's main element is a ball the player should control and carry to a remarked target position in the scene. To control such ball the player should touch and swipe the device's screen inputing a direction and speed he wants the ball to take. Once the ball is positioned at target, scene ends, a score is given to user and next scene is unlocked.

Fig. 2. Balloma's scene sample. Elements of scene are: a) The ball (blue), b) Floating elements (green), c) Score records (red), d) Target position (yellow)

While the user carries the ball to target he can make the ball go trough floating elements in the scene which makes the score increase. Also a timer is included in the scene which influences the final score, lower scene's completion running times provides higher rewards.

Figure 2 presents the game's first scene which appears unlocked by default. In Figure 2 the ball is bounded by a blue box. The blue diamonds bounded with a green box appears floating in the scene, if reached by the ball makes the score (upper left red box) increase. Target position is remarked inside a yellow box. Also there is a timer (lower left corner red box) that influences the final score.

IV. Environment

An interface is implemented to provide a RL suitable environment by which the agent can interact with the game to construct the optimized policy. Since the game runs in Android, the environment should provide a controlling interface with an android physical or virtual device. Such functionality can be attained through the Android Debug Bridge (adb[3]).

In this proposal the states are represented by raw scene frames of 240x240. It means at each time step t the environment is at state $s_t \in \mathbb{R}^3$ formatted as a tensor of shape $(240, 240, 3)$. A similar approach was presented in [1] and [4], however scenes in those works are 2D spaces and in this case it is 3D. Frames are cropped to exclude score indicators and focus on more important pixels.

On each time step the agent can chose to take an action $a_t \in \mathbb{R}^4$. Actions are presented as a tuple $a_t = (x_0, x_1, y_0, y_1, \nu)$ containing swipe attributes: *a)* x_0 starting swipe x coordinate, *b)* x_1 final swipe x coordinate, *c)* y_0 starting swipe y coordinate, *d)* y_1 final swipe y coordinate and *e)* swipe velocity ν.

Each episode starts with the first swipe inputted by the agent. If the balls falls or get to target position environment is set into a terminal state. Also it will be constrained in time to avoid long episodes.

V. Reward function

At each time step, after the agent choses an action a_t and such action is applied to environment, a reward r_t is observed. Such reward is computed taking into account two coefficients: *a)* the ratio of gathered floating diamonds over all such elements in the scene(ω) and *b)* episode time passed since it started (φ). Then reward is calculated as:

$$r(s_t, a_t) = \frac{\omega_t}{\varphi_t} \tag{6}$$

Such coefficients appears as elements of the scene at a fixed frame position. It should be cropped and processed so as to convert such pixels into a suitable representation of its digits. In order to extract the score on each time step from those scene elements, in this work it will be trained a digit recognizer on the well known MNIST handwritten digit database [6].

[3]https://developer.android.com/studio/command-line/adb

A. Your first solution

Describe your first solution here.

B. Your second solution

Describe your second solution here.

C. Your third solution

Describe your third solution here.

1) Subsubsection Heading Here: Use the subsubsection command with caution—you probably won't need it at, but I'm including it this an example.

VI. Criteria for Assessing Solutions

This may be a modified version of your proposal depending on previously carried out research or any feedback received.

VII. Research Methodology

The main difference between this section and the one in your report proposal is use of verb tense: there you suggested what you will do and here you will describe what you did. Be concise and precise when outlining how you researched your potential solutions. Remember that your research should be guided by:

- Relevance to the context of application
- Your assessment criteria
- Practicality

So it may be worth commenting on your research methodology in light of the above (e.g., justifying a particular approach).

In this section, only describe how you collected data, and explain what you did to test your criteria. *Do not include your findings in this section.*

VIII. Analysis and Interpretation

In this section you will mainly analyze your data in terms of your assessment criteria; e.g., do the data suggest that a particular solution is "cost effective" "environmentally acceptable", "technically feasible" or "affordable"?

Be logical and selective when analyzing/interpreting your research data. For example, if a proposed solution is proven to be far too expensive to realistically implement in your context, is there any value in discussing whether it is "culturally viable" or "technically sustainable"? Perhaps in this case you can focus more attention on solutions that your research suggests are more valid. Do not just throw huge quantities of raw data at your reader and leave them to interpret it. Present enough to transparently support any conclusions you draw and make sure that you offer justifications for your analysis.

Be honest and reflective while discussing your data. Your data might be too limited or unclear to interpret with accuracy—explain this, perhaps suggesting how this shortcoming could be addressed. Admitting the above will help you draw more honest and worthwhile conclusions.

Remember that research is an imperfect and ongoing process that should be open to question and verification. Therefore, unless convinced by the absolute strength of your evidence, you should be tentative in your language choice when interpreting/analyzing research results. Selectively use *hedging* (language which indicates a lack of certainty) to modify the tone of your analysis and any conclusions that result from this.

	Growth Media				
Strain	1	2	3	4	5
GDS1002	0.962	0.821	0.356	0.682	0.801
NWN652	0.981	0.891	0.527	0.574	0.984
PPD234	0.915	0.936	0.491	0.276	0.965
JSB126	0.828	0.827	0.528	0.518	0.926
JSB724	0.916	0.933	0.482	0.644	0.937
Average Rate	0.920	0.882	0.477	0.539	0.923

TABLE I. Some impressive numbers

Here are some examples that show differing degrees of certainty:

- it appears that ...
- it can be tentatively concluded that ...
- it is almost certain that ...
- perhaps the evidence indicates ...
- this seems to point to the fact that ...
- this could be interpreted as evidence of ...
- without doubt its application would prove beneficial for ...

Finally, don't introduce any new content (e.g., research methods or solutions) within this section—this will prove confusing for the reader. The reader should clearly understand that you are, based on specific criteria, interpreting the results of your research in order to test the viability of various solutions to remedy a particular problem. The sole function of this part of the report is to openly discuss your research findings in order to set up your conclusions/recommendations.

A reference to Table I.

IX. Conclusions and Recommendations

Conclusion shows what knowledge comes out of the report. As you draw a conclusion, you need to explain it in terms of the preceding discussion. You are expected to repeat the most important ideas you have presented, without copying. Adding a table/chart summarizing the results of your findings might be helpful for the reader to clearly see the most optimum solution(s).

It is likely that you will briefly describe the comparative effectiveness and suitability of your proposed solutions. Your description will logically recycle language used in your assessing criteria (section VI): "Solution A proved to be the most cost effective of the alternatives" or "Solution B, though a viable option in other contexts, was shown to lack adaptability". Do not have detailed analysis or lengthy discussions in this section, as this should have been completed in section X.

As for recommendations, you need to explain what actions the report calls for. These recommendations should be honest, logical and practical. You may suggest that one, a combination, all or none of your proposed solutions should be implemented in order to address your specific problem. You could also urge others to research the issue further, propose a plan of action

or simply admit that the problem is either insoluble or has a low priority in its present state.

The recommendations should be clearly connected to the results of the report, and they should be explicitly presented. Your audience should not have to guess at what you intend to say.

APPENDIX A
WHAT GOES IN THE APPENDICES

The appendix is for material that readers only need to know if they are studying the report in depth. Relevant charts, big tables of data, large maps, graphs, etc. that were part of the research, but would distract the flow of the report should be given in the Appendices.

APPENDIX B
FORMATTING THE APPENDICES

Each appendix needs to be given a letter (A, B, C, etc.) and a title. LaTeXwill do the lettering automatically.

REFERENCES

[1] Lillicrap, Timothy P., et al. "Continuous control with deep reinforcement learning." arXiv preprint arXiv:1509.02971 (2015).

[2] Konda, Vijay R., and John N. Tsitsiklis. "Actor-critic algorithms." Advances in neural information processing systems. 2000.

[3] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. 2014. Deterministic policy gradient algorithms. In Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32 (ICML'14), Eric P. Xing and Tony Jebara (Eds.), Vol. 32. JMLR.org I-387-I-395.

[4] Mnih, Volodymyr, Kavukcuoglu, Koray, Silver, David, Rusu, Andrei A, Veness, Joel, Bellemare, Marc G, Graves, Alex, Riedmiller, Martin, Fidjeland, Andreas K, Ostrovski, Georg, et al. Humanlevel control through deep reinforcement learning. Nature, 518(7540):529–533, 2015.

[5] (2011) Bellman Equation. In: Sammut C., Webb G.I. (eds) Encyclopedia of Machine Learning. Springer, Boston, MA

[6] LeCun, Y. Cortes, C. (2010), 'MNIST handwritten digit database', .