The Learning Triangle

Software Requirements Specification
For unsupervised learning algorithms

Version 1.2

Date	Version	Description	Author
31.10.2016	1.0	First set up	LearningTriangleTeam
27.11.2016	1.1	Added new Use Case Diagram	LearningTriangleTeam
12.12.2016	1.2	Changed Use Case Diagram (Scope)	LearningTriangleTeam

Introduction

Purpose

The Learning Triangle project wants to create a generic algorithm which is able to learn behaviours in a generated world with non-changing rules. In this project we want to learn how unsupervised machine-learning is working. Therefore this system will not provide any interaction with other services of the user himself. It is just to research possibilities using machine-learning algorithms.

Scope

Our scope is visible in our Use Case Diagram. You will find it in this document.

Acronyms

Acronyms	Description
None	So far

References

Reference	Source
Tensorflow	Tensorflow

Overview

Overall Description

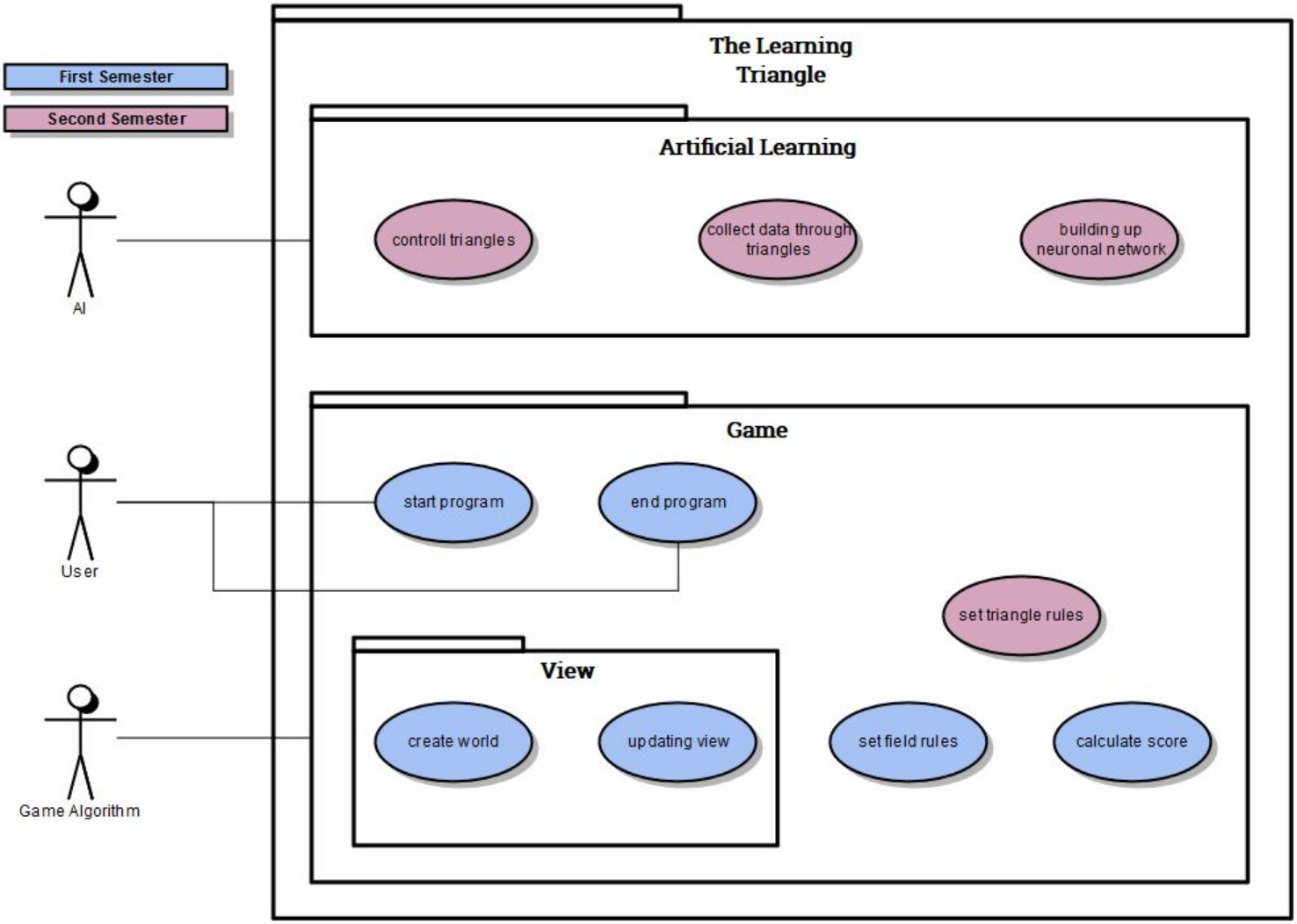
The Vision

As planet Earth started evolving, a fortunate sequence of events led to the development of a friendly environment which made life possible. Such as comet-showers out of water in a huge mass. As a result of the Earth's positioning in the habitable zone in our Solar System, a

world with suitable prerequisites for single- and multi-cell microorganisms evolved. Organisms, stimulated by their surroundings, developed through mutual bacterial symbiosis. Thus, complex behavioral patterns arose. Creatures learned to survive.

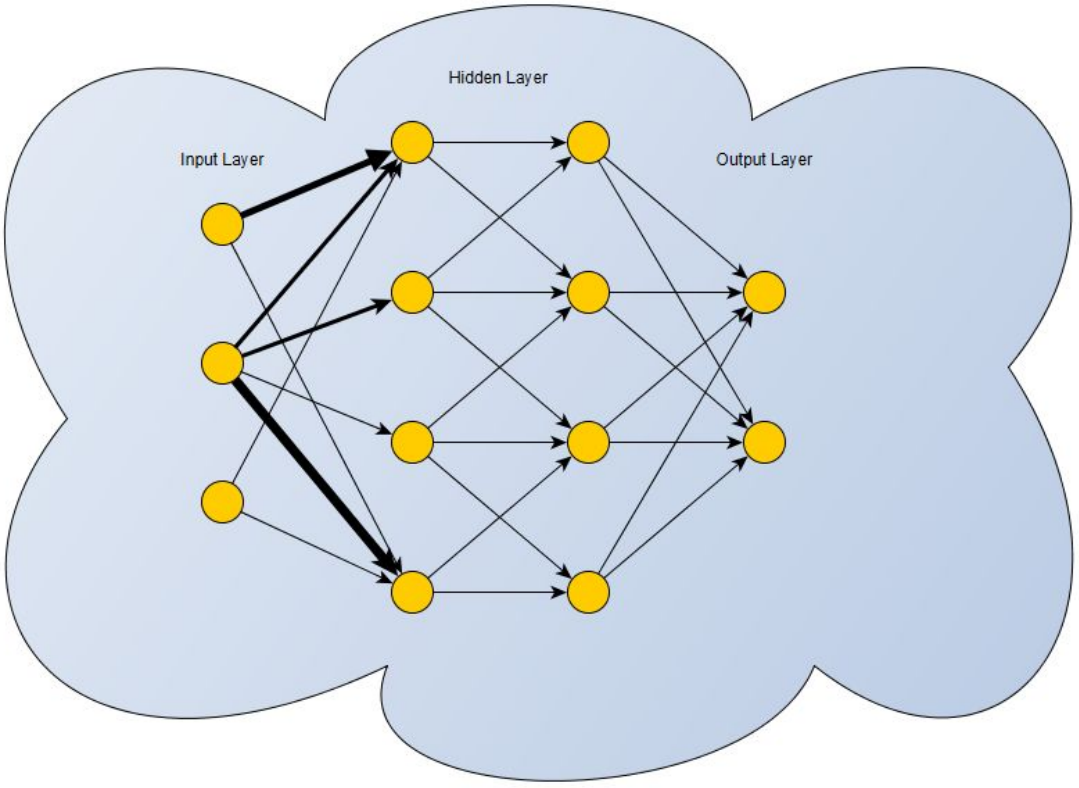
Similar to this procedure the project “The Learning Triangle” is intended to be a simulation of artificial creatures in a generated world. To archive this target we want to use the stand of the art technology for machine-learning. We want to build a neuronal Network that can learn different behaviors and patterns by itself. Therefore we want to do research on AI in general and monitor creatures patterns evolving. We are in the opinion that this particular area of computer-science will change the world within months and years.

First Idea of our Use Case Diagram



As seen in the diagram, our user isn't able to do many things. Our AI controlls the game, but not the rules. They are set by our game algorithm, which displays the game, its objects and the score. This will be clarified during our progress of development.

Example of a neuronal network



Ein neuronales Netzwerk besteht aus einem sogenannten Input-Layer, einem Hidden-Layer und einem Output-Layer. Der Input-Layer repräsentiert die Eingabe. Der Hidden-Layer besteht aus mehreren Schichten weiterer Neuronen. Im Hidden-Layer werden Beziehungen zwischen Input- und Output-Layer ausgearbeitet. Der Output-Layer ist das Ergebnis der verschiedenen Transformationen. Alle Neuronen sind Stufenweise unterschiedlich stark untereinander verbunden. Dies wird anhand Input-Layer und 1. Hidden-Layer beispielhaft dargestellt. Wie stark die unterschiedlichen Neuronen miteinander verbunden sind, hängt von den bereits erlernten und trainierten Daten ab.

A neuronal network consists of an Input-Layer, a Hidden-Layer and an Output-Layer. The Input-Layer is like the name already says the Input. The Hidden-Layer is the complex part of this network. Many layers of neurons create a connection between each other and also between Input- and Output-Layer. The picture shows different bold arrows, which symbolizes connections of different strenghes.The strengh of the connections will be determined by the trained data. This data will come from the “game” developing and moving forward.

Specific Requirements

Requirement	Description
Deeplearning4J	Deeplearning4Jis needed to create the neuronal network what is the base of the learning algorithm.
JUnit 4.0	JUnit is to maintain stability of the code. JUnit is a testing framework for Java
Mockito	Mockito is a framework that adds more functionality to the JUnit module. It can “mock” classes and functions. That means we can create artificial function calls without the need of creating the whole environment.
Maven	Maven is a dependency manager. We use this to provide all requirements in just one file without the need of configuring the buildpath over and over again.
Maven-Surefire	Maven-Surefire makes sure all defined tests will run. Only in case all tests pass its functionality the project will build.

Functionality

Generation of different Overworlds using seeds

The artificial world of the creature can be created from a seed. For same seeds the overworld will be the same for testing different behaviours on same ground. A overworld consists of the following fields:

1. **Energy**
   Energy is used from creatures to sustain themselfes
2. **Instant-Death**
   Instant-Death is a field that kill all creatures stepping over it
3. **Poison**
   Poison wound creatures in a way, that they losing energy faster
4. **Wall**
   Wall is a unpassable field for a creature
5. **Normal**
   Here does nothing happen

Living creatures

Living creatures are the naming learning triangles that follow specific rules:

1. They lose energy by moving forward
2. They have to move forward every time
3. They can turn
4. They can reproduce themselves using energy
5. They can run faster

The goal of the creatures are to move as much distance as possible. The distance is the result of distances of every creature summed together.

Learning algorithm

The learning algorithm consists of a neuronal network created by Google's Tensorflow. The network for the creatures have to learn the patterns of the artificial world around them.

Usability

Using our program isn't very hard to achieve, the one important thing a user must know is how to start applications. This application will ask for a world-seed what will create the overworld of the creature that is living in. After that, the creature begins to "live" in the environment and the user can see how the creature is figuring out in which way the world is functioning.

Reliability

It is important that while running our program, a connection to the server is necessary. ...

Performance

Supportability

The tools and frameworks we use are well known and very important. They will be well supported in the future.

Design Constraints

This project will follow the MVC-Pattern.
We also want to use a test driven development method. The test driven development follows the rule:

On-line User Documentation and Help System Requirements

During the run of our application there won’t be many questions how to use it, so the more important thing is that the user must understand the game mechanics. Our main goal is to design our “game” as understandable as possible. That means that we will use game elements and a design that is self-explaining. Also a guide will be available in the application to answer upcoming questions.

Purchased Components

n/a

Interfaces

User Interfaces

Hardware Interfaces

Software Interfaces

Communication Interfaces

Licensing Requirements

n/a yet

Legal, Copyright and Other Notices

n/a yet

Applicable Standards

n/a yet

Supporting Information

You can find our blog in the intenet:
https://thelearningtriangle.blogspot.de/
Also you can look at our project on Git:
https://github.com/Noixes/thelearningtriangle