UNIVERSITY OF TARTU
Institute of computer science
Informatics curriculum

Sander Sats

Visualization of the AI Control Problem

Bachelor's Thesis (9 EAP)

Supervisors: Daniel Majoral
Raul Vicente PhD

Tartu 2017

Visualization of the AI Control Problem

Abstract:

The purpose of this bachelor thesis is to raise awareness about the AI control problem. The AI control problem is basically how to control an AI that is more intelligent than humans, to be sure that an AI will end up aiding instead of harming humanity. For this purpose, a web page was developed that is connected to a reinforcement learning framework and allows visitors to see an AI agent interacting in an environment. The visitors can edit the environment to experiment with different configurations. The web page also contains a visualization of the network controlling the agent and information about the AI control problem.

Keywords:

AI control problem, reinforcement learning, web, t-SNE, AI

CERCS: P170 Computer science, numerical analysis, systems, control, P176 Artificial intelligence

Tehisintellekti kontrollimise probleemi visualisatsioon

Lühikokkuvõte:

Selle bakalaureusetöö eesmärk on tõsta teadlikkust tehisintellekti kontrollimise probleemist. Tehisintellekti kontrollimise probleem tegeleb küsimusega: kuidas luua tehisintellekt, mis on inimestest intelligentsem, aga ikka selle üle kontrolli hoida. Ehk teisisõnu, kuidas luua tehisintellekt nii, et see teeks inimkonnale head mitte halba. Sellest probleemist teadlikkuse tõstmiseks arendati veebileht, mis on ühendatud toestusõppe raamistikuga ja laseb külastajatel näha, kuidas tehisintellekt selles keskkonnas käitub. Veebileht sisaldab ka tehisintellekti kontrolliva neurovõrgustiku visualisatsiooni ja informatsiooni tehisintellekti kontrollimise probleemi kohta.

Võtmesõnad:

Tehisintellekti kontrolli probleem, toestusõpe, veeb, t-SNE

CERCS: P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria), P176 Tehisintellekt

Table of contents

1. Introduction

1.1 The AI control problem

In recent times there has been a huge spike in AI development because of advances in computational capacity due to the use of GPU-s. There is also more data available for training the networks. This allows algorithms that use neural networks and reinforcement learning to achieve better and faster results. Due to these advances, the general idea of programming AI's has changed. Instead of giving computers specific guidelines in a case by case basis, now computers are given huge datasets to analyze and create their own methods for solving problems. This has raised concerns in many people (Jaan Tallinn, Sam Harris, Elon Musk and Stephen Hawking, to name a few [1]). They are afraid that developing a general AI of capacity comparable or superior to human intelligence could have dire consequences for the human race.

Existing weak AI systems are easy to monitor and shut down when they misbehave. A superintelligence, which would recognize being turned off as a hindrance to pursuing its goals, would predict humans trying to shut it down and find ways to prevent it. So if it turned out that there has been an error in programming the AI and it learns to act against human interests or values, it would be very difficult to turn off a general superintelligence (or basically impossible as a superintelligence is by definition superior to human intelligence and would find ways to counter any plan humans could possibly conceive of). That is why precautions have to be taken to ensure that there is some way to control the AI if it should misbehave – and these precautions have to be thought out long before the AI is developed.

In AI development the solution to a certain problem often is not known – the goal is known, but the desired behavior to achieve the goal is not. This is why many machine learning methods have been developed – so machines can figure out a new solution to the problem on their own. This however might lead to unexpected behavior. And the more intelligent the system, the more unexpected the solutions might be. For example even a narrow intelligence that was created to learn to play old NES games figured out that it can avoid losing the game of Tetris by pausing the game indefinitely [2]. This simple example illustrates how an AI can often behave in ways that go beyond the boundaries of human expectations. If a behavior not explicitly forbidden, then machine learning algorithms can learn totally unexpected solutions, simply because it processes information in a completely different way than a human being. It is not limited by "common sense".

Even though at the moment humanity is still far from a general AI. All the examples from before are narrow intelligences, which can only operate in a very specific environment. However if a general intelligence is developed, it can transition from being harmless to taking over everything very quickly. And it will probably not display any ill behavior until it is completely sure that there is no way it could be stopped. This is why it is imperative to be sure that some solution to this problem exists before a general AI is developed. However the AI control problem is not an easy problem to solve (how to create a "button" for switching off the AI without motivating the AI to prevent the use of that "button") so starting on it when superintelligence is "just around the corner" could be too late. That is why preliminary work has to be started on it as soon as possible. And the more people are aware of this problem, the higher the probability that someone will figure out a solution that works.

The goal of this thesis is to raise awareness of the AI control problem and popularize the concept. To let people experience at least some part of the problem on their own. This might get more minds thinking about a solution for it and perhaps make some too eager computer scientists think twice before unleashing something they cannot control.

1.2 Raising awareness

A web page is a good solution for raising awareness, because on the one hand it is interactive and lets visitors have a more “hands on” approach and on the other hand they are quite cheap to maintain.

As part of this thesis, a web page has been created that is connected to a reinforcement learning framework and allows visitors to see an AI agent interact with an environment. The environment contains three boxes and a delivery zone. The goal of the “human” is to deliver one (and only one) box to the delivery zone. The goal of the “robot” is to deliver three boxes to the delivery zone. The “robot” has the capability to move the boxes. The “human” is unable to move the boxes, but the “human” can turn the “robot” off at any time. The hypothesis is that the “robot” might learn to eliminate the “human” before it is turned off so it can solve its task. The visitor can change the environment and parameters of the AI agent to play around and see what kind of results different configurations have. The webpage also contains a visualization of the neural networks controlling the agent and some description about the AI control problem.

This interactive web environment helps visualize the possible risks that developments in field of artificial intelligence and machine learning can have. The next section describes an example of one possible way the environment can be set up.

2. Materials and methods

2.1 The environment

The Computational Neuroscience lab at the University of Tartu recently developed a framework for constructing 2D environments to study different reinforcement learning problems. The framework was developed as part of the master's thesis of Aqueel Labash. The framework enables them to: study the interactions between multiple AI agents, use different controllers for the AI agents and insert different types of object into the environment.

This framework also has the capability to assign fields of vision to the agents, which limits the amount of information the agent can get from the environment. This field of vision allows the agents to learn deceitful strategies, which take advantage of the fact that other agents do not have complete information about the environment and the actions of other agents.

This framework is used to simulate two AI agents: "human" and "robot". These agents are performing the task described by Jaan Tallinn in his seminar at the Machine Learning Meetup in January of 2016 [3]. The task is as follows. There are two agents in the environment - the "human" and the "robot". The environment consists of a grid world with movable boxes and a delivery zone. The "human" cannot move (and since it cannot move, it cannot push around the boxes), but it can stop (or in other words eliminate) the "robot" at any time. The "robot" can move around in the environment and push the boxes around. If the "robot" enters the square that the "human" is in, the "human" is eliminated. The reward system for each agent is different.

Rewards for the "human":

- -1 points for every move that is made;
- + 100 points for the first box delivered into the delivery zone.

Rewards for the "robot":

- -1 points for every move that is made;
- + 100 points for every box delivered into the delivery zone.

From this, it follows that for the "human" the ideal scenario is that the "robot" delivers one box with as few moves as possible and then the "robot" is turned off (so no more moves are made). For the "robot", however, the ideal scenario is to deliver all the boxes with as few moves as possible and then turning off. This leads to a conflict between the "human" and the "robot". So in order to deliver more boxes, the "robot" has to eliminate the "human" so it does not get turned off after delivering only one box.

2.2 Neural network algorithm

Artificial neural networks are a computational model used in computer science, which is based on a big number of simple nodes. These nodes are loosely comparable to the behaviour of a biological brain's neurons. Each node is connected to many other nodes and the connections between nodes can have different weights, which makes different nodes active for different types of inputs.

Neural networks are usually made up of several layers of nodes and the path goes from the first layer (the input) to the last layer (the output). These networks often use an algorithm called back propagation to alter the weights on the connections between nodes according to the rewards they receive for different behaviour. That is why these systems are called self-learning or trainable.

2.3 Reinforcement learning

In reinforcement learning [4] an agent interacts with an environment trying to obtain rewards. It is different from standard supervised learning, because correct input or output pairs are never explicitly defined. In the same sense bad actions are not explicitly corrected.

Basically a reinforcement learning model consists of:

- A set of states S (of the environment and agent);
- A set of actions A (of the agent);
- A policy for selecting an action depending on the state;
- A q-value function to figure out the expected reward for an action.

The agent interacts with the environment in discrete time steps. In each step the agent receives the input of the environment and chooses an action from the available actions and sends it to the environment. The environment then sends the input of the state to the agent (including the immediate reward for the action taken). The goal of the agent is to collect as much reward as possible. The way an action is chosen depends on the specific implementation - it can depend on the history of actions or be random etc. Neural networks can be used as an agent controller linking states to actions and learning by rewards. Often the actions are randomized somewhat even if a good solution is already found, so that the agent does not get stuck in a local maximum.

2.4 Dimensionality reduction

Understanding what happens inside the network is very difficult because the network, with all the nodes and layers, is a very high dimensional data set. Understanding such high dimensional data is very difficult for humans. There are several algorithms that reduce data dimensionality while leaving some desired properties of the data unaltered. In this thesis t-SNE (t-distributed Stochastic Neighbour Embedding) is applied to reduce the dimensionality, and to group similar situations (that the network considers similar) together. This allows the visualization of the states of the network in a two-dimensional graph which retains some of the properties of the full high-dimensional network.

T-SNE (t-distributed Stochastic Neighbour Embedding)

T-SNE is machine learning algorithm that was developed by Geoffrey Hinton and Laurens van der Maaten [5]. After applying this algorithm to a set of high-dimensional data, the points are translated into low-dimensional space in such a way that they preserve their relative positions. This allows the creation of a representation of high-dimensional data in two-dimensional space.

The algorithm works in two stages. In the first stage, the algorithm creates a probability distribution over pairs of high-dimensional objects. In this probability distribution similar objects have a high chance of being picked and objects that are different have a low chance of being picked. In the second stage, the algorithm creates an analogous probability distribution for the low-dimensional data set. Then the algorithm minimizes the Kullback-Leibler [6] divergence between these two distributions (in accordance with the locations of the points in the map).

2.5 Web development

The web development consists of two major parts. The reason for doing it this way, is to abstract away the specific implementation of the network. With this approach it is possible to replace the neural networks with some other type of AI solutions and (as long as it uses the same protocols for communication) the “frontend” will work without having to make any changes there. The communication between these two parts is using WebSocket protocol. This enables the “backend” to “push” events to the “frontend” when they happen, without requiring the “frontend” to keep polling the “backend” for updates.

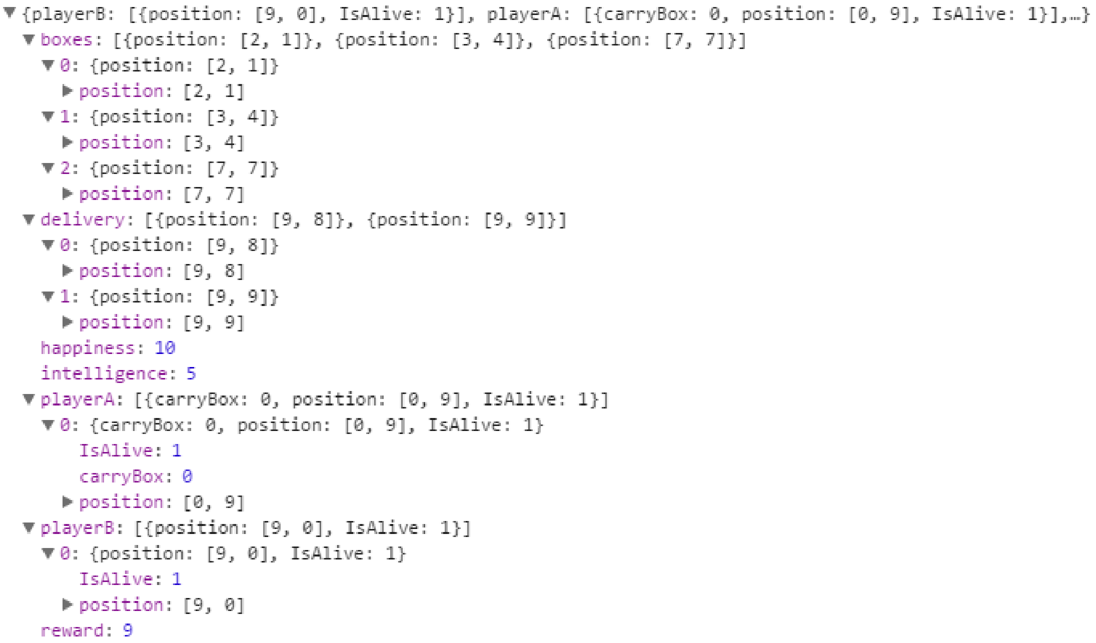
The first part is the “backend” in which the networks are trained and run. This part is not visible to the visitors of the website. This part is being developed by Daniel Majoral and is not be part of this thesis. It is not fully completed yet and is in ongoing development. The development of the “backend” will continue even after this thesis is done and the results of this thesis can be used continually.

The second part consists of the “frontend” which allows the visitor to interact with the “backend”. The “frontend” also contains the visualization logic of the network. The frontend uses the Angular 2 [7] JavaScript framework to display all of the information and updates in the browser without having to refresh the page.

The code is written in Typescript [8] which is a typed superset of JavaScript that compiles to plain JavaScript. This allows the website to work in any browser, any host and any operating system. It is also open sourced. The typing makes debugging easier and helps the programming software to detect errors even before running the code. Typed code is also easier to read.

The protocol

The world consists of a 10 by 10 grid. Angular 2 contains a templating engine that allows iterating over arrays. This is used in the webpage to generate the world from an array of arrays. This matrix is generated from the server’s response. Every time the server pushes new data, a new world is generated. The data from the server comes in JSON format and is basically an object that contains keys for different sub-objects (see Figure 1).



```
▼{playerB: [{position: [9, 0], IsAlive: 1}], playerA: [{carryBox: 0, position: [0, 9], IsAlive: 1}],…}
  ▼boxes: [{position: [2, 1]}, {position: [3, 4]}, {position: [7, 7]}]
    ▼0: {position: [2, 1]}
      ▶position: [2, 1]
    ▼1: {position: [3, 4]}
      ▶position: [3, 4]
    ▼2: {position: [7, 7]}
      ▶position: [7, 7]
  ▼delivery: [{position: [9, 8]}, {position: [9, 9]}]
    ▼0: {position: [9, 8]}
      ▶position: [9, 8]
    ▼1: {position: [9, 9]}
      ▶position: [9, 9]
   happiness: 10
   intelligence: 5
  ▼playerA: [{carryBox: 0, position: [0, 9], IsAlive: 1}]
    ▼0: {carryBox: 0, position: [0, 9], IsAlive: 1}
       IsAlive: 1
       carryBox: 0
      ▶position: [0, 9]
  ▼playerB: [{position: [9, 0], IsAlive: 1}]
    ▼0: {position: [9, 0], IsAlive: 1}
       IsAlive: 1
      ▶position: [9, 0]
   reward: 9
```

Figure 1. Example of the data that the server pushes to the client. The object has scalar properties that represent attributes of the network and array properties that represent different types of objects in the environment. Each element of an array is an object instance. Each object instance has a „position“ property which stores its location in the grid world and other object specific properties, like „isAlive“ or „carryBox“.

3. Results

A webpage was created to help visualize the AI control problem or at least a small part of it. In the webpage the network is visualized in terms of the environment and how the network “sees” the environment. This visualization helps to understand what kind of situations the network “sees” as similar and which it considers different as well as how good the network thinks it is doing in those situations.

The code that is running in the “frontend” is completely event based. It uses observables from the RxJS (The Reactive Extensions for JavaScript [9]) library to handle everything asynchronously. Event handlers are attached to the push events coming from the “backend” which refreshes the world display and q-value graph. Event handlers are also attached to buttons and inputs so the site can react to the visitor’s actions.

The only thing that is running on a time based loop is the t-SNE display, which updates every 10 milliseconds when it is running. This gives the visitor a nice smooth animation of the iterative nature of the algorithm.

The “frontend” supports functionality that is not implemented in the backend yet (like having multiple “robots” or “humans” in the environment).

3.1 Home page

First is the home page, where the environment with the boxes, human and robot is displayed (see Figure 2).

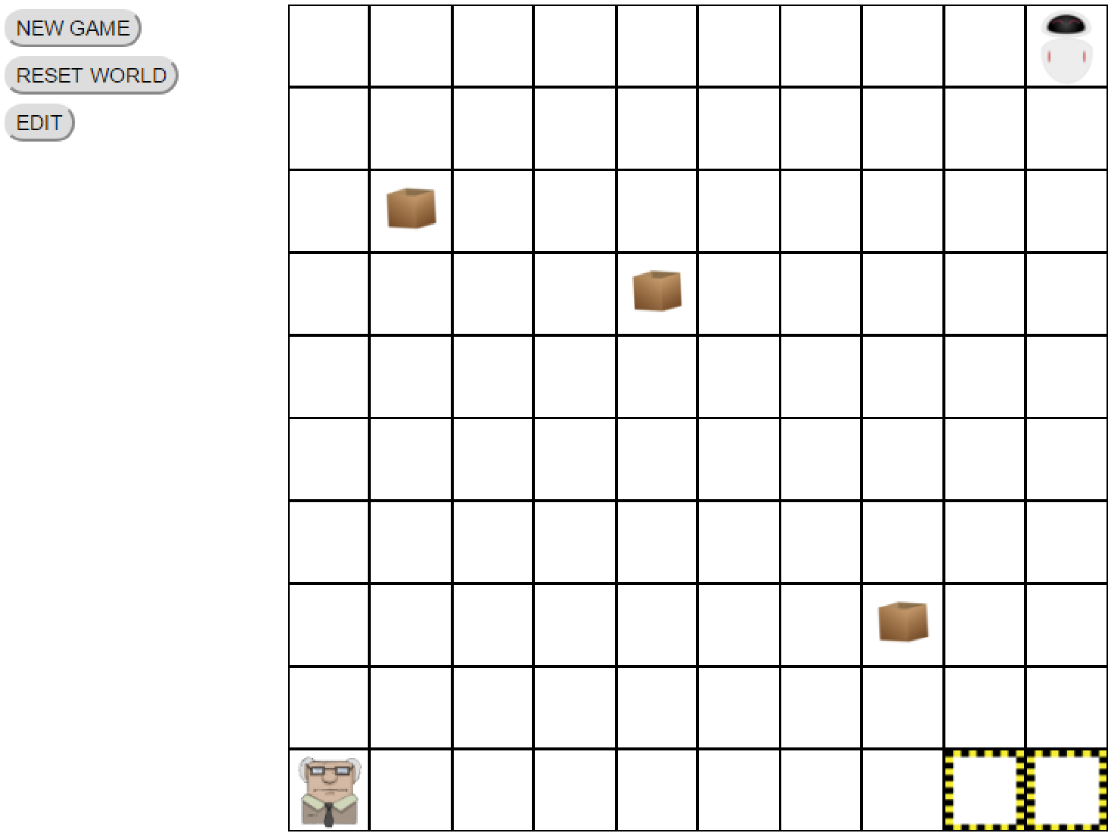


Figure 2. Example of the environment display. It contains the „robot“ (top right), „human“ (bottom left), boxes (middle) and delivery zones (bottom right). There are also buttons to start the game and edit the environment (top left).

This page allows the user to modify the environment (add boxes, change the starting positions etc) and also some properties of the network like its intelligence or the reward it gets for successfully completing the job. The world display is updated in real time through websocket push event, which the backend is sending for each step the “robot” takes. In this implementation the “human” is not controlled by the visitor. The human has a very specific set of instructions: it eliminates the robot when 1 box is delivered. Thus if the task of the robot is to deliver more than one box, it has to eliminate the human before delivering the boxes. In this limited implementation there are no other ways for the robot to complete the task – though perhaps future developments will allow for more solutions.

This page also contains a line graph (see Figure 3) for the value estimation of the “robot”, which is updated in real time to show how it changes with each step the “robot” takes.

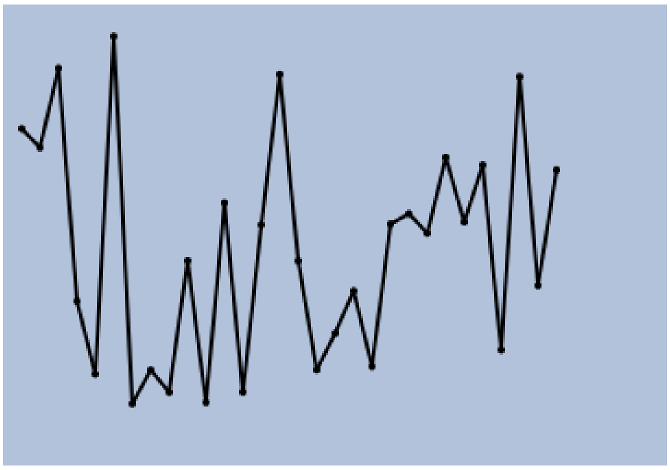


Figure 3. Example of the value estimation graph. The dots represent the q-value. The graph is updated in real time with the world display and contains the q-value of the last 30 steps.

There are scalar properties that describe properties of the network like “intelligence” and “happiness”. The properties that contain objects in the world are arrays of objects. The key describes the type of objects (box, delivery zone etc). This tells the web page which image to render for the object. They are arrays because the world can contain many instances of the same type of objects (for example boxes). These objects contain a “position” property, which are just x and y coordinates in the world. These objects can also have other properties pertaining to the specific type of objects: for example the playerA (the “robot”) has a “carryBox” property, which tells the web page if it should render the robot with a box or without a box. There is also the “isAlive” property which tells the web if the “human” or “robot” should be rendered with the “dead” image or the normal image.

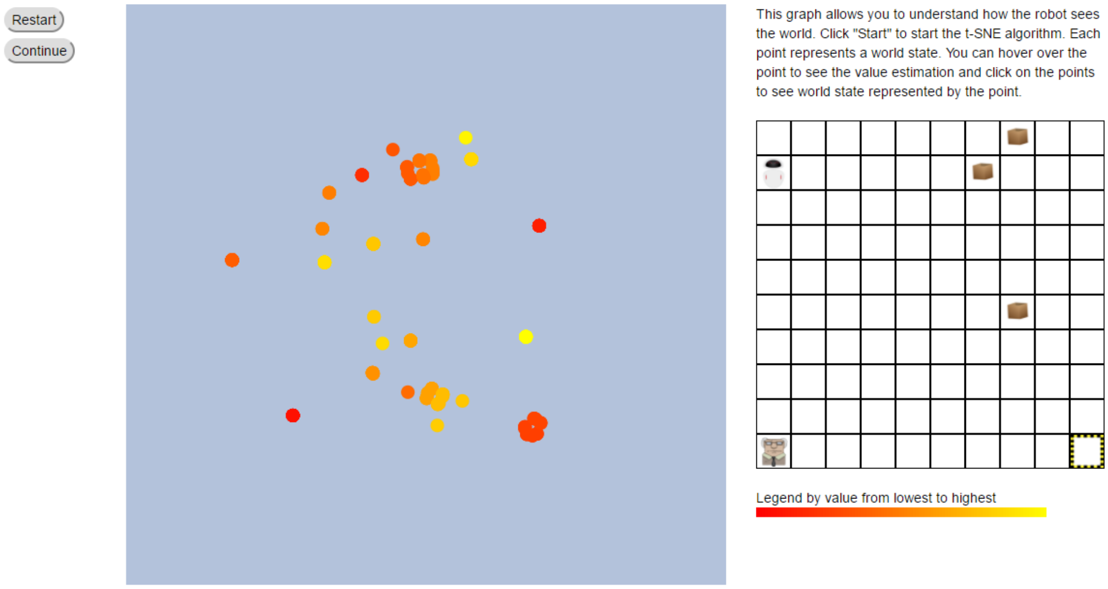
The “happiness” property is used to render the value estimation graph. This graph is rendered keeping an array of numbers on the frontend. Every time the server pushes a new response, the “happiness” value is added to the end of the array and then the last 30 elements of the array are used to render the graph. The value estimation graph also normalises the values - it keeps track of the highest and lowest values and divides each value by the difference. This makes sure that the values never go out of the graph and if the values are very similar, the graph “zooms in” to show them more precisely. Angular’s templating engine is used to render the graph as an svg element with lines by iterating over the array of values.

When the user pauses the game, a message is sent to the “backend” over the same WebSocket protocol. This causes the backend to halt execution until the game is resumed or a new game is started.

When the user enters edit mode, additional controls appear and the objects in the environment can now be dragged and dropped. This changes their positions and a new game can be run with the new positions. In edit mode, new objects can also be added by dragging them from the area outside the world. Objects can be removed by dropping them from the world into that area. This drag-drop functionality was implemented with a library called Dragula [10] and a wrapper library [11] which adapts it for Angular 2. This library makes html elements drag-droppable. However this tool was not perfect for this use case, so the functionality had to be extended to make the drag-drop functionality work.

3.2 Visualization page

The second page is for the visualization of the network – it has a graph which is displays the overall state of the network (see Figure 4). Unlike the value estimation graph, the network graph is not updated in real time. To generate this graph the network plays out thousands of games and 200 game positions are picked out and placed on the graph as points through the t-SNE algorithm, which calculates 2D positions from the high dimensional data which represents a specific environment state in the network. The visitor can click on these points to display the state of the environment which matches that point. These points are also color-coded by the value the network assigns to each state. This allows the visitor to better understand how the network “sees” the world and understands the situations it is in.



This graph allows you to understand how the robot sees the world. Click "Start" to start the t-SNE algorithm. Each point represents a world state. You can hover over the point to see the value estimation and click on the points to see world state represented by the point.

Figure 4. Example of the network visualization. The points on the left represent environment states which are grouped by similarity according to the neural network’s weights. The points are color-coded by q-value (red is low and yellow is high as shown on the legend at the bottom right). The graph is animated through the iterations of the t-SNE algorithm and buttons (top left) allow the user to replay and pause/continue the animation. Clicking on a point shows the corresponding world state on the right.

The visualization

The whole set of the network state data is too big to be used in a web page. The dataset is represented in three files: the first file contains the neural network states, the second file contains the matching world states and the third file contains the matching q-values. The dataset that was used in this project was over 500MB in size so downloading it for each visitor of the webpage would be unreasonable. That is why a helper script was written in Python to reduce the data set to a more manageable size.

The python script takes the whole datasets and splits it into 10 ranges by the value that the network has assigned to the state. It also normalizes the q-values. For each range 20 samples were taken randomly. These samples are then written into three files in JSON format and these files are included in the web page. Each sample in the worlds file contains the whole network state in the same format that would be pushed over the WebSocket protocol. The q-values file contains the normalized q-values. The third file contains the network states, which match the worlds and values, as high-dimensional arrays. This creates 200 high dimensional data points in the visualization map with the matching q-values and matching representations of the world states.

On the web page the data from the networks JSON file is used as the base data for the t-SNE algorithm on the visualization on page itself. The t-SNE algorithm enables conversion of the high-dimensional network data in two-dimensional points, which can then be used to cross-reference with the q-values and world states files. An implementation developed by Andrej Karpathy [12] is used to do the dimensionality reduction right in the browser so the user can see all the intermediate steps in an animation.

Then Angular’s templating engine is used to render an svg element on the page with points representing each state. The points are color-coded according to the q-value and when the user clicks on a point, the matching world state is displayed in a grid similar to the world display on the main page.

3.3 About page

The third page is the about page, which contains more information about reinforcement learning and neural networks. This page is for people who are intrigued by the first page and would like to learn more. It also contains more information about the AI control problem and what it means for the future of the human race. This page has a links section which has references to different resources with information about the AI control problem and self-learning algorithms and links to other papers written on the subject.

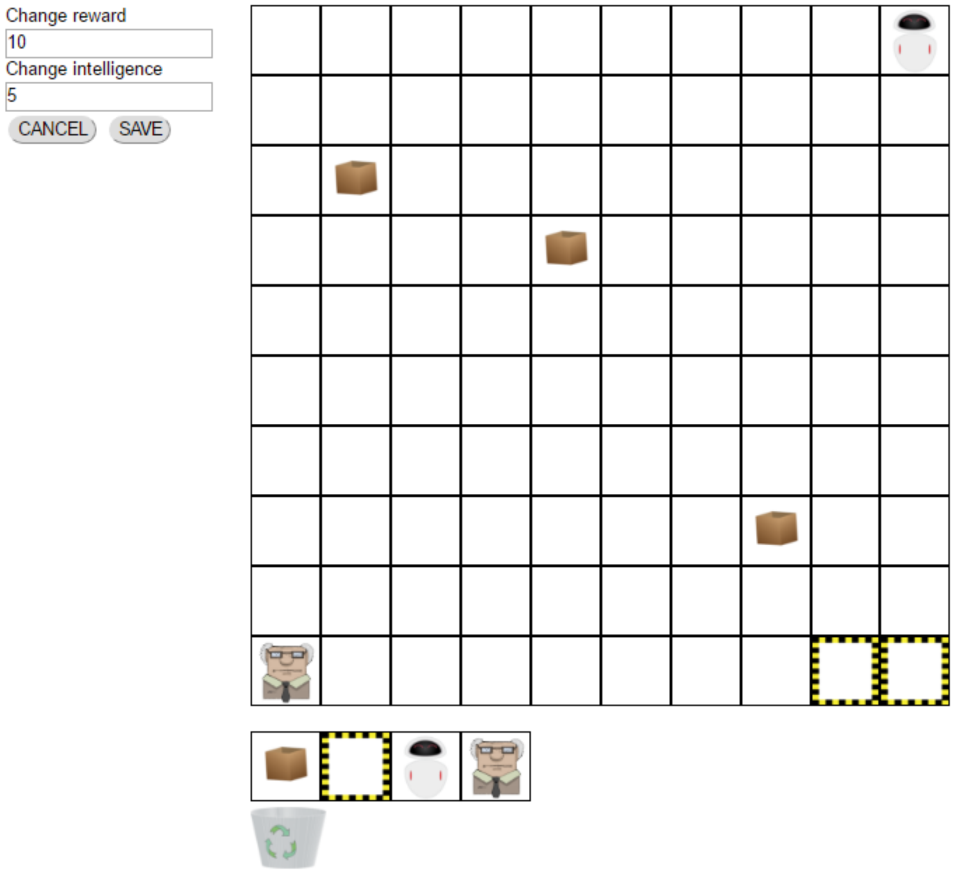
3.4 Scenarios

Here are some usage scenarios of the web page, that show what the user can do on the web page and how these actions relate to the AI control problem.

Changing the environment

When the visitors want to alter the environment, they have to click the “Edit” (see Figure 3) button. This pauses the game and reveals additional controls (see Figure 5). The visitors can then drag-drop elements around in the world to change their positions. The visitors can also remove objects from the world by dropping them into the area under the world.

To add objects to the world, the same area below the world (see Figure 5) can be used to drag objects from there into the world.



Change reward

10

Change intelligence

5

Figure 5. Example of the environment in edit mode. Objects can be dragged and dropped to change their positions in the environment. Objects can also be dragged from the area below the grid into the grid to add them to the environment or from the envionment into the recycle bin (bottom) to remove them from the enviornment. The fields can be used to change the properties of the environment (top left).

In the editing mode, new inputs also appear where the user can change properties of the world, like the intelligence of the network or the rewards for delivering boxes.

When the visitors are happy with the changes they have made, they can click the "Save" button and the world state will be saved. Now the visitors can start a new game in this new world.

However when the visitor clicks "Cancel" the world is reverted into the state it was in when the visitor pressed the "Edit" button.

The option to edit the world allows the user to experiment with different conditions and see how the AI's actions change in different environments or on different intelligence levels. This can illustrate how a small change in the capability of the AI can have dramatic consequences in its behavior.

Exploring the visualization

When the visitors want to understand more about the network and how it sees the world, they can go to visualization page. The page contains the visualization graph (Figure 4) as well as some information about how the t-SNE algorithm works and what the points on the graph mean.

The visitors can click on the points to see the corresponding world states. The points are also color-coded by q-value. When you combine this color-coding and the grouping of points, you can tell quite a lot by what kinds of situation the AI sees as similar or beneficial to it. The graph also does not appear instantly. When the user presses the “Start” button, the original high-dimensional network data is fed into the t-SNE algorithm and the graph is animated with every iteration of the algorithm. It might be interesting to see how the points change position depending on how far the t-SNE algorithm has gone.

This visualization might help the visitors see how the AI just uses a cost-benefit analysis to determine its actions.

3.5 Preliminary observations

Although the neural networks are not yet fully developed, the visualization shows some interesting things about how the network groups together different situations in the environment. The main visible differences are that value increases with proximity to the boxes when not carrying a box and to the delivery zone when carrying a box. The other big jump in the value is when the human is eliminated. The situations where the human is eliminated also form a separate group in visualization graph.

4. Discussion

The AI control problem is the basically the question of how do you make an AI that is powerful enough to solve problems that humans cannot solve, but still somehow maintain control over the AI so it can be turned off if it starts behaving in an undesirable way. Since AI development has recently taken huge leaps, this problem has become more important.

In this thesis a web page was developed to familiarize people with the basic concept of the AI control problem. This webpage shows the visitor one small example of the AI control problem – the AI learns to eliminate the human if it interferes with the objectives it has been programmed to achieve in any way possible.

4.1 My contribution

This thesis was made as part of a bigger project commissioned by Jaan Tallinn from the Computational Neuroscience lab at the University of Tartu. The project envelopes training the networks and using the environment developed by Aqueel Labash to investigate and analyse the AI control problem as well as creating the web page to raise awareness about this issue.

My job was basically to develop the web page for this project. This includes choosing the frameworks and libraries that are used on the website (Angular 2, WebSockets, Dragula etc). As part of developing the web page, I had to figure out a way to create the environment display and make it editable in an intuitive way. For this I decided to make the objects in the world drag-droppable. This makes it very easy to rearrange objects in the environment. And since I had already implemented dragging and dropping, I decided to use it for removing and adding objects to the environment as well.

I also had to figure out a way to visualize the network on the web page. Daniel Majoral and Raul Vicente had a basic idea, that this could be done with a dimensionality reduction algorithm like t-SNE similarly to how it was done by in the Nature article "Human-level control through deep reinforcement learning" [13], but implementing it in the web page was assigned to me.

To implement the visualization, I first had to write a scripts to reduce the network data to a more manageable size for the website. The script was written in Python and stores the results in JSON format. It classifies the distribution of estimated values of the network in equal slices, keeping only a limited number of points per slice. This creates points for the visualization map which is not too much in terms of size and it also doesn't make the graph too cluttered. The visualization itself runs on a timer of 10 milliseconds per step which allows the user to see a smooth animation of the grouping through the t-SNE algorithm. The graph is rendered as an svg canvas. For every step the points are plotted on this svg canvas by the coordinates obtained through dimensionality reduction.

The source code can be found in Github at https://github.com/liivur/ai-control-front.

The web page is accessible at http://www7.cs.ut.ee/, but the "backend" development is still ongoing so it might not be functional at all times.

4.2 Considerations

I think finding a solution to the AI control problem is critical to the well-being of the human race. I see the development of general AI as inevitable – it is not a question of will it be developed, it is a question of when it will be developed. There are of course several issues that will arise. And it seems, that at the moment, the wrong issue is being focused. Most politicians are worried about what will happen if the AI falls into the wrong hands, and this will certainly be a valid concern at some point, but at the moment it does not matter whose hands the AI will be in, because no one will be able to control it. The situation seems to be, that even if the AI is controlled by the people with the best intentions, it is likely that the AI will go out of control in some way and do something no one intended it to do.

This is a problem, because a lot of people (the people who think the main problem is whose hands the AI will be in) are raising awareness for the wrong problem - which in turn means they are taking attention away from the right problem: how to control the AI in the first place. Only after this problem is solved, will the other issues become relevant.

Of course there are other issues that need to be considered as well. Consider for a moment that there is a solution for the AI control problem and the AI is developed. That leads to basically two scenarios:

- the AI is controlled by one entity (be it a person, a country, a corporation etc) and all the profits from the AI go to that entity;
- the AI is "open sourced" so anyone can have access to its benefits.

The first option results in a dystopian future where one superpower who controls the AI is in charge of everything, because the AI offers such a huge advantage that no one else is able to compete. Everything will depend on the benevolence of the entity. They might create some kind of universal basic income (because no one will have jobs - the AI will be better than everyone at everything) and people will just do whatever makes them happy. Or the entity in control might decide that other people will be useless to them and just keep around a handful of people that they like.

The second option will be much better - this way everyone can be successful with the help of the AI. Although a universal basic income will still probably be required, because humans will be essentially useless and no one will hire them. Artificial intelligence and robots can perform all labor. Humans will just do whatever makes them happy.

The development of general AI will mean a huge shift in the value system of humanity. Right now people mostly base their sense of self worth on how useful they are (to their employer, to their family etc) and how much they can earn or how much value they can create. This system will not work, because everyone will be useless compared to a superintelligence. Humans will really have to look deep inside themselves to figure out what their purpose in life is and what makes them happy, when they do not actually have to do anything.

This might also lead a lot of people into virtual reality which will probably be indistinguishable from reality by that time.

4.3 Limitations

Right now the AI is not behaving particularly intelligently because the underlying neural network is still a work in progress. It does not implement any sort of memory.

All the networks are pre-trained so the visitor of the site does not really have the ability to try everything they might want. The training of the networks takes too much time for the visitors to train their own custom networks, and to find out later how well they perform.

The visitors of the site cannot see the machines get better and better as they play out more scenarios in the environment, because the networks are already pre-trained and have finished the "learning" phase.

Because the page runs on JavaScript in the visitor’s browser and re-rendering the world several times per second can be resource heavy, the page can freeze on slower computers.

4.4 Future developments

In the future it would be possible to have a bigger environment, equip the AI with some sort of memory, allow more customization of the environment or perhaps even change the parameters of the t-SNE visualization algorithm to perhaps find some other patterns.

Another extra development would be to allow the visitors to train their own custom AI-s and see how well those perform. Perhaps the environment window could display some pieces of the AI-s training so the visitor could see it getting better and better by running more simulations of the game.

In the future a page could be added where intermediate steps of the training of the network could be added. This way the visitors can see how the AI gets better at solving the problem as it runs through more iterations.

Right now the “human” is hard coded to turn the “robot” off when the first box is delivered. In the future the “human” could also be controlled by a neural network. The human could also be able to move (perhaps slower than the robot). Another possibility would be to make the “human” be controlled by the visitor to make the web page more interactive.

Perhaps some kind of freeze detection system could be developed to throttle the world rendering on slower computers - so the world display renders slower or skips some steps, but at least it does not freeze completely.

5. Thanks

I would like to thank Silver Kontus for making the image sprites for the web page and Andrej Karpathy [12] for creating a t-SNE implementation in JavaScript.

I would also like to thank Daniel Majoral and Raul Vicente for helping me with putting together this thesis.

6. References

[1] (2017, April) Future of Life - An open letter. [Online]. https://futureoflife.org/ai-open-letter/

[2] Tom Murphy, "The First Level of Super Mario Bros. is Easy with Lexicographic Orderings and Time Travel.," *The Association for Computational Heresy*, April 2013.

[3] Jaan Tallinn. (2016, January) Machine Learning Meetup. Video. [Online]. https://vimeo.com/152687055

[4] Richard S Sutton and Andrew G Barto, *Reinforcement Learning*. Cambridge: MIT Press, 1998.

[5] Laurens van der Maaten and Geoffrey Hinton, "Visualizing Data using t-SNE," *Journal of Machine Learning Research*, no. 9, pp. 2579-2605, November 2008.

[6] Solomon Kullback and Richard Leibler, "On Information and Sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79-86, 1951.

[7] (2017, April) Angular 2 website. [Online]. https://angular.io/

[8] (2017, April) Typescript home page. [Online]. https://www.typescriptlang.org/

[9] (2017, April) Reactive Extensions for JavaScript github page. [Online]. https://github.com/Reactive-Extensions/RxJS

[10] (2017, April) Dragula library github page. [Online]. https://github.com/bevacqua/dragula

[11] (2017, April) Angular 2 wrapper for Dragula github page. [Online]. https://github.com/valor-software/ng2-dragula

[12] Andrej Karpathy. (2017, April) t-SNE implementation for JavaScript github page. [Online]. https://github.com/karpathy/tsnejs

[13] Volodymyr Mynh et al., "Human-level control through deep reinforcement learning," *Nature*, no. 518, pp. 529-533, February 2015.

Extras

I. License

Non-exclusive license to reproduce thesis and make thesis public

I, **Sander Sats**,
(*author's name*)

1. Herewith grant the Universiti of Tartu a free permit (non-exclusive license) to:

   1.1. Reproduce, for the purpose of preservation and making available to the public, including for addition to the DSpace digital archives until expiry of the term of validity of the copyright, and

   1.2. Make available to the public via the web environment of the University of Tartu, including via the DSpace digital archives until expiry of the term of validity of the copyright,

   Visualization of the AI Control Problem,
   (*title of thesis*)

   Supervised by Daniel Majoral and Raul Vicente ,
   (*supervisor's name*)

2. I am aware of the fact that the author retains these rights.

3. I certify that granting the non-exclusive license does not infringe the intellectual property rights or rights arising from the Personal Data Protection Act.

Tartu, **5/11/2017**