**Smarts- Explorations in a Virtual World**

Smarts is a small world that allows explorations in reinforcement learning. To begin to understand how the program works, start the program using the instructions on the previous page, and then work your way through the following tutorial.

Clicking on the Smarts link should download and run Smarts. If it asks about a certificate say 'yes'.

When you first open Smarts, you'll see the main bar:

![Main Bar](image)

and the world window:

![World Window](image)

The green dots are plants and the red critters are, well, critters. They have numbers in gray that identify them.
To see the critters in action, click on the step button several times. The critters will do one of several actions: move forward, turn to one of 6 orientations, or rest. If they move onto a plant they will automatically eat it and be reinforced for doing so. They also gain health points by doing so. In fact, health points (positive or negative) are the only reinforcement in this version of Smarts. Running into rocks (which will be gray dots on the screen) subtracts health points and acts as a negative reinforcement.

If you click the 'run' button the critters will move around and attempt to learn by reinforcement what is good and what is bad. After 100 time steps the simulation will pause.

After a while of clicking on the run button you may notice that some of your critters have died. They do this a lot. Smarts can be a harsh, cruel world and only the faster learners survive. This is the first of many metaphors that link Smarts to the real world.

If all your critters have died, click on the 'Restore World' button to make them come back to life.

How can we make our critters last a little longer? Just like in real life, longevity and knowledge is a combination of choosing the right actions, learning from past reinforcements, picking the right combinations of tried-and-true methods and novel approaches, and a little luck. Smarts includes a learning algorithm that builds in these ideas into a simple learning process that can be a little hard to understand. In the beginning, let's just watch them learn.

Smarts allows editing of the world and the parameters that govern the learning process. Click on the Edit World button to get this window to appear:
Let's try a different world. Click on the Load Predefined World pop-up menu and select world #1:
This world has plants and rocks. The critters lose one health point for every time step, and 10 health points for every rock they run into. They gain up to 80 points for each plant they eat, to a maximum of 100 health points. Before you run this world, click on the Edit Critters button to make this window come up:
If you click on an item in the list, this critter is highlighted in the Critter Editor and a yellow border is drawn around that critter in the world window:
Smarts lets you save critters that have learned. Run this simulation until just a few critters are left. Make sure you pause it before all the critters have died. If this happens, reset the world and try again.

In my simulation, three critters survived:
The critters turn white as they get older, which is another unfortunate relation between Smarts and the real world.

These three critters have survived, so it must mean that they learned something where the others did not. We can look at their memories, although it may not mean much at first. To see a graphical representation of their memories, make sure the enable editing button is not checked in the world editor, and then click on a critter to bring up its neural network window:
Later we'll go through what the various fields mean, but basically the large red and green squares imply some form of learning and memory. Let's save this critter to test it out in a different environment. Open the critter editor if it is not already open. Click on the name of the critter you'd like to save:
(critter 12 in my case) and then click the save critter button. This will save the critter's memory to a file once you name it and remember where you saved it.

Now let's see how your trained critter does in a different world. Will his knowledge generalize? Let's try world #4. Open the world editor window again and select world 4:
This is a harsh world that has lots of food but also lots of rocks to bump into. Try running this world a few ties. Very few critters survive very long!

Now let's see if the critter you trained in the first world can use his knowledge in this new world. Restore the world, and click on the Edit Critters button to open the critter editor. Highlight one of the critters and click on the load critter button:
Find your critter file you saved with the first world and select it. Now when you click on each critter in the world, you'll see that one knows a lot, and one doesn't know anything yet:
Critter 0 contains the knowledge from the first run, while Critter 1 doesn't know anything. Click the step or run button to watch what happens in this new world. Lots of things can happen, but you may find that critter 0 moves more aggressively toward food in the beginning.

If you reset the world, you'll have to reload the critter's memory again (Matt is working on a snapshot button to solve this).
**Training your critter to survive.**

To see how harsh this environment is, reset the world and run it several times. Your critters will most likely die off fairly rapidly. Let's provide a little encouragement for them to learn at a faster rate.

Restore the world and then click on the world editor button to bring up this window:

<table>
<thead>
<tr>
<th>World Editor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Objects to change...</td>
</tr>
<tr>
<td>☑ Enable Editing</td>
</tr>
<tr>
<td>Learning parameters...</td>
</tr>
<tr>
<td>Discount Rate: .7</td>
</tr>
<tr>
<td>Reinforcement schedule...</td>
</tr>
<tr>
<td>Bump: -5</td>
</tr>
<tr>
<td>Run properties...</td>
</tr>
<tr>
<td>Number of Steps in a Run: 100</td>
</tr>
<tr>
<td>World properties...</td>
</tr>
<tr>
<td>☑ Maintain Critters</td>
</tr>
<tr>
<td>How long plants live: -1</td>
</tr>
<tr>
<td>Critter properties...</td>
</tr>
<tr>
<td>☑ Two Layer Network</td>
</tr>
<tr>
<td>Save and load worlds...</td>
</tr>
<tr>
<td>New World</td>
</tr>
<tr>
<td>Save and load snapshots...</td>
</tr>
<tr>
<td>Load Snapshot</td>
</tr>
</tbody>
</table>

NOTE: All options in RED will perform an action that amounts to clicking on Restore World. This will potentially wipe out any changes to the World, such as Critters and their memories, and the addition or removal of objects (Rocks, Critters, Plants) to the World. It is recommended that you use Save World before changing any RED options.

To help the critters learn, let's change the reinforcement for Move and Turn to 10:
Important: Once you have changed the values, hit the Apply button to make your changes understood.

Now when you run the simulation you'll find that the critters tend to move more, and when they reach the end of the path they turn more. But they might get addicted to turning, because they get reinforced for it. Thus they may spend all their time turning. Let's wean them off turning. Pause the simulation and change the value for Turn to 1. Hit Apply and resume the simulation. Now your critters should move more, and get more food in the process.
Let's wean them off the move reinforcement as well. Change the values for Turn and Move to 0 and apply the changes. Hopefully we've given the critters enough encouragement early on in learning so that they can now survive on their own.

With some tries, you should get two critters that zip up and down collecting food and staying healthy. Look closely at their behavior. Are they turning the same direction each time they hit the end of the path? Critters can actually turn around without turning left or right first. Do both critters do the same thing?

We can see the results of the learning by looking at the memory windows. Make sure the Enable Editing button isn't checked and then click on each critter to bring up their neural networks. Here are mine:
The exact values are a little hard to interpret, but it is clear by comparing the two memories that the two critters have solved the problem in different ways.

**World Editing**
In Smarts, you can create your own worlds to test your trained critters. Open the World Editor and click on one of the radio buttons at the top of the window:
Now when you click a location on your world, a plant will be added. Using plants, rocks and other critters you can create worlds that test different reinforcement schedules and training regimens. To save your world (complete with critter locations), click the Save World button, and click on the Load World button to load a saved world. The New World button lets you start from scratch.

**Propagating the Smartest Critters**

Now that you understand how Smarts works, let's try a little selective evolution. If you have been editing a world and you want to keep it, click the save world button and save your world as a file. That way when you click the Reset World button your changes will be saved. Start a new world by clicking on the Reset World button and let the simulation run until you have 2 or 3 critters left. These are the ones who have learned; the rest have died under the brutal conditions that Smarts worlds sometimes possess. But the strongest
have survived, and we can now begin to propagate their memories to create a new breed of supercritters. Pause the simulation if it is not already stopped. Then open the critter editor and find what you think is the best critter. Select its name and click on the Save Critter button:

A dialog box will come up and you can save your critter. Type a filename in and click 'Save'.

Let's start over again, but this time with better critters. Click the Restore World button to bring back all your dead critters. Like any good reincarnation, however, the critters have lost their memories. We have the ability, however, to transplant good memories into the empty vassals that are the current critters. Open the Edit Critters window and select a critter. Then click on the Load Critter button and select your critter file that you saved previously. This critter will now contain all the memories that the old critter did, and is presumably much smarter than the other critters right now.

If you want, you can then click on the Clone Memory button:
This will take the memories from the current critter and copy it to all the other critters. This is helpful if you want to share the learning from the best critter to those that have not learned yet.

This process of finding the best critter and propagating its memory to the others can be repeated as long as you like. Each time you are giving the best memories (which really are mappings between situations and actions) more chances to grow and thrive. Ultimately you should be able to train a critter that can survive in many different environments.

**Etcetera**

This is a first snapshot of Smarts. There are lots of subtle things we are still discovering, and it is a rich environment to test ideas about interactions between an organism and its environment, as well as how the environment dictates learning along with the reinforcement schedule. This tutorial does not introduce the learning algorithms behind Smarts, but future materials will. We also will explain how to interpret fields in the neural network window, and what positive and negative reinforcement imply.

For now, explore Smarts and evolve your own memory!