Learning to Plan with Logical Automata

Brandon Araki[1]*, Kiran Vodrahalli[2]*, Cristian-Ioan Vasile[1], Daniela Rus[1]
[1]Massachusetts Institute of Technology
[2]Columbia University
*Equal contributors

Imagine you are an alien visiting earth and you need to learn how to drive. Not only do you have to learn how to drive a car, but you also have to learn the rules of the road.

How would you learn these things? One way to do it would be to watch other people driving. You would observe that people stop at red lights, drive on the right side of the road, and try to not run into each other or off of the road.
You would be learning *the rules of the road*. Once you have learned these rules, you would hopefully be able to get into a car and drive on the streets safely.
A crucial element of your learning is that you would conceptualize the rules separately from the minute, low-level actions of the cars. You would have a compact, discrete representation of the rules in your head that is easily *interpretable*. One of our goals in this work is to learn interpretable policies.

Interpretable

- The structure of the learned policy is grounded directly in meaningful interpretations

Let's say that you observe a bad driver, perhaps an AWOL self-driving car, run a red light.

As an alien, you might think to yourself, "Oh, since cars decide to run a red light 1% of the time, I should too." A police officer should be able to tell you, "NEVER run a red light!"

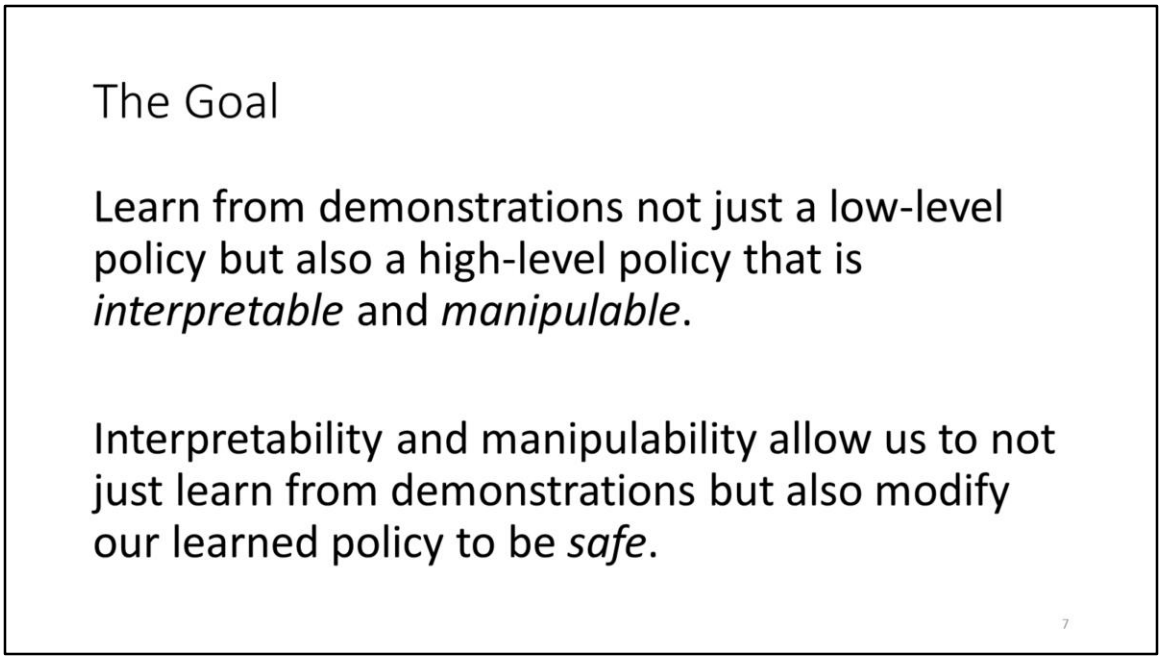
This should be possible because the representation of the rules in your head is *manipulable*, and can therefore be made *safe* if you learn from faulty experts.

Manipulable

- A human operator can easily modify the learned policy to perform similar but different policies.
- Manipulable ≠ Composable

6

Manipulability is different than compositionality. Compositionality involves combining policies to form new policies, whereas manipulability means that we can directly and easily modify the learned policy, which changes the behavior of the agent.



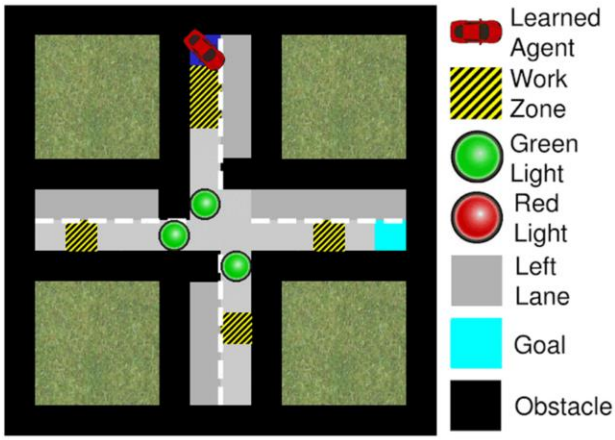
To restate: our goal is to learn from demonstrations both low-level action-based policies as well as a high-level control policy that is interpretable and manipulable.

Learning in an interpretable and manipulable manner, with the aim of learning a safe controller, is desirable for more than just aliens. Rules govern many parts of human life and also exist hidden in data.
They are in video games, sports, financial markets, household chores, cooking, disease diagnosis, and human interactions, just to name a few areas where rules are important.

Case Study: Driving

Case Study: Driving

Rules:
- Avoid obstacles and work zones
- Avoid left lane when possible
- Do not run red lights
- Reach the goal



9

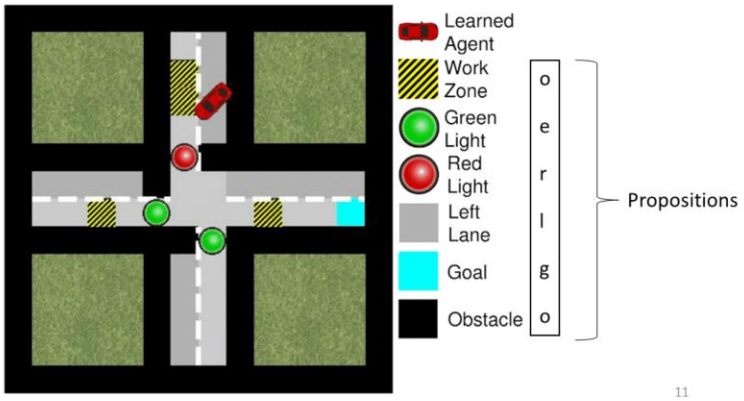
Here is our simplified driving environment. The agent follows rules such as “avoid work zones and obstacles”, “prefer the right lane”, and “stop at red lights”.
How can we learn these rules from demonstrations such as the one on this slide?

Case Study: Driving

- Assume rules can be encoded as a Finite State Automaton (FSA)
- Assume relevant features of the environment can be detected
- Assume number of FSA states is known
- Assume environment outputs current FSA state as well as low-level state
- Learn the transitions between FSA states as well as low-level transitions

Case Study: Driving

- Propositions: logical True/False variables

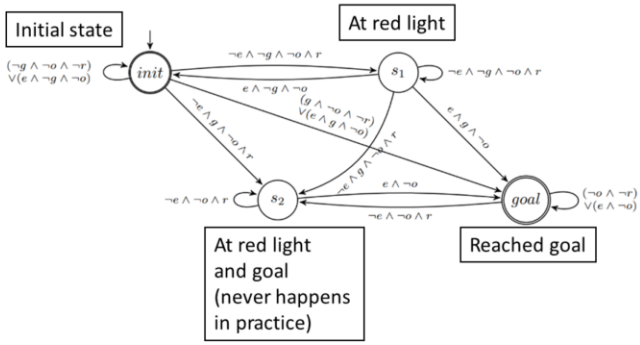


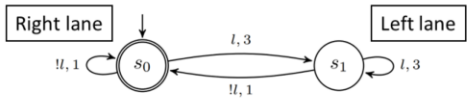
11

'Features' correspond to propositions in a logic formula

Case Study: Driving

- Ground-truth FSA has 9 states (represented here as two separate FSAs)



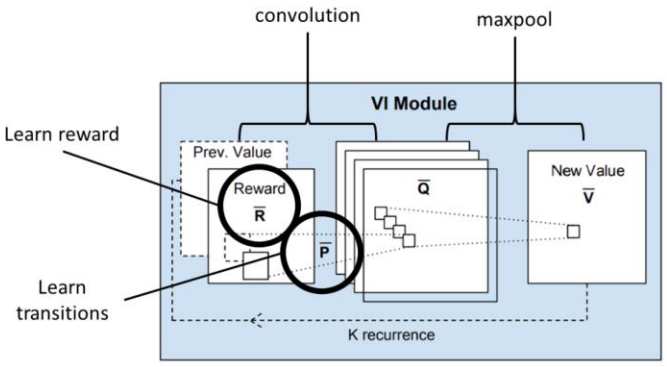


12

Transitions between states are controlled by the truth statements of propositions

How Does It Work?

Value Iteration Networks (VINs)



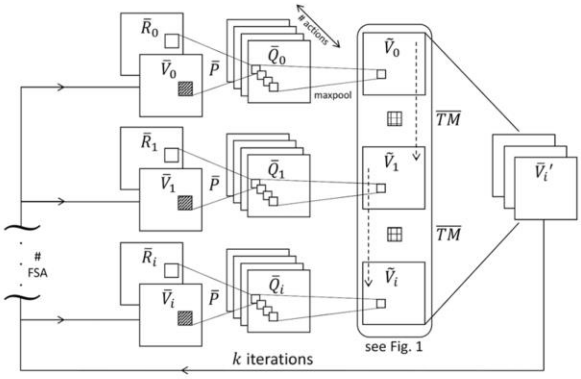
Tamar, Aviv, et al. "Value iteration networks." *Advances in Neural Information Processing Systems*. 2016.

14

Our model is based off of the Value Iteration Network architecture. VINs put value iteration into a fully differentiable architecture, where the Q function is calculated using a convolution and the value function is extracted using a max pool operation. This makes it possible to learn the reward and transition functions of the environment.

However, it also imposes a few limits on the system - for example, the environment has to be able to be interpreted as a grid-like structure so that it can be encoded as a matrix and then have convolutions applied to it.

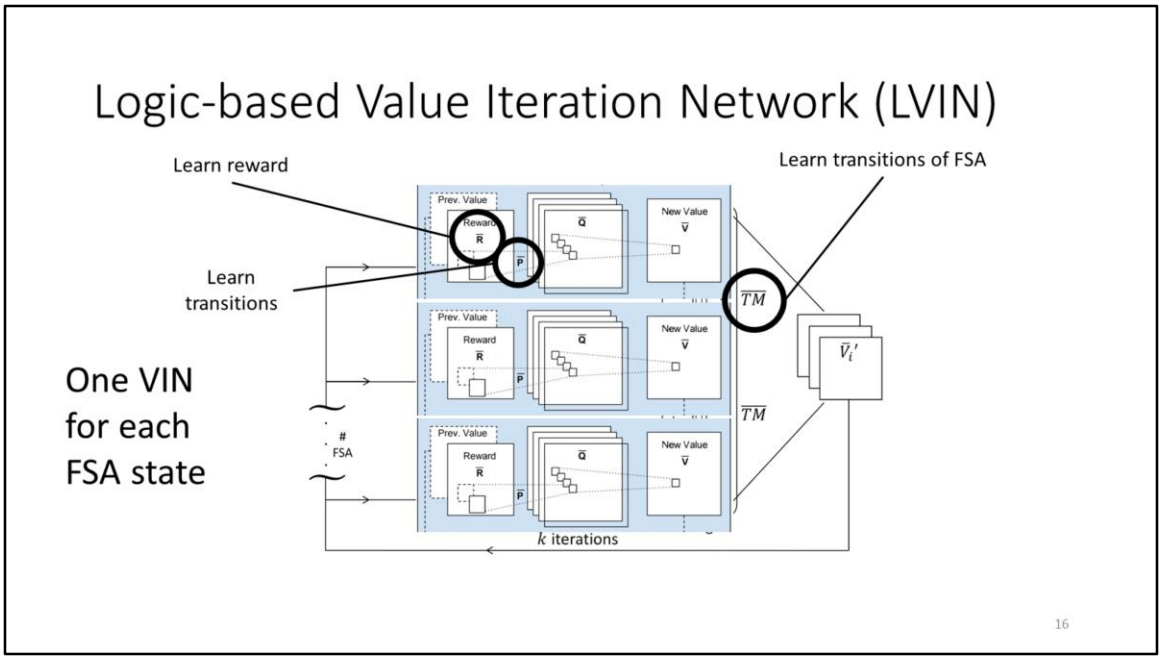
Logic-based Value Iteration Network (LVIN)



15

Here is a representation of our system, which we call LVIN, for Logic-based Value Iteration Network.

There are two main differences between LVIN and VIN. One is that we essentially have a separate VIN module for every FSA state in the environment.

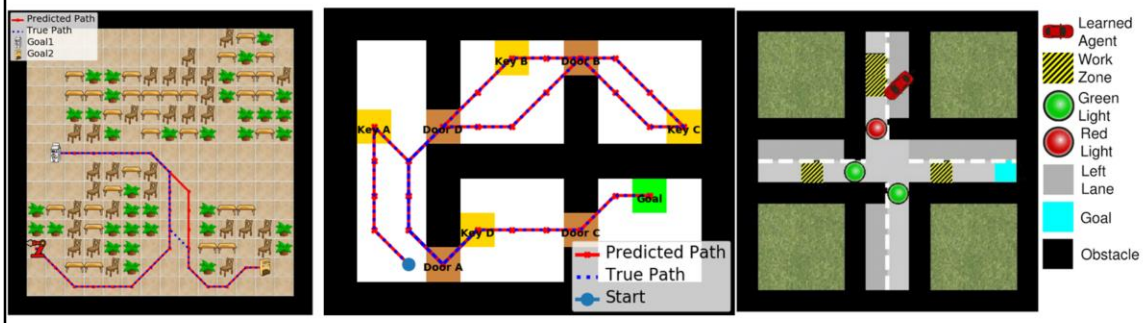


You can see here, I've tiled the VIN architecture on top of LVIN.
Doing this allows us to learn the reward and transition functions of each FSA state. In addition, we apply a second convolution on the value function. Learning this convolution allows us to learn the FSA structure.

What Makes LVIN Different?

- Interpret the high level of a hierarchical model as a FSA / logical specification
  - *Interpretable*
- Incorporate the FSA into value iteration so that changes to the FSA result in changes to the policy
  - *Manipulable*

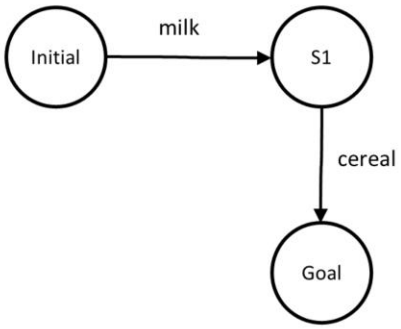
Experiments

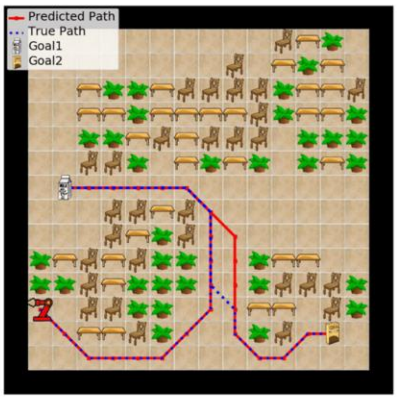


18

Case Study: Kitchen

"Pour milk into bowl, then pour cereal"

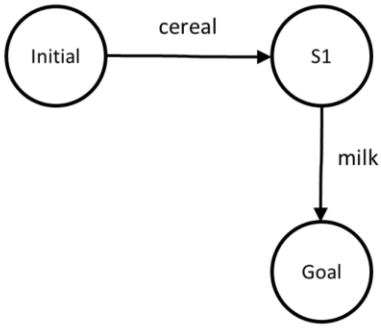




19

Case Study: Kitchen

"Pour cereal into bowl, then pour milk"

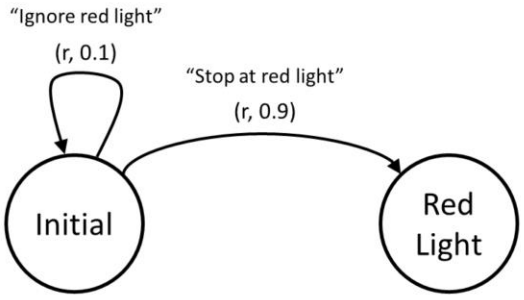


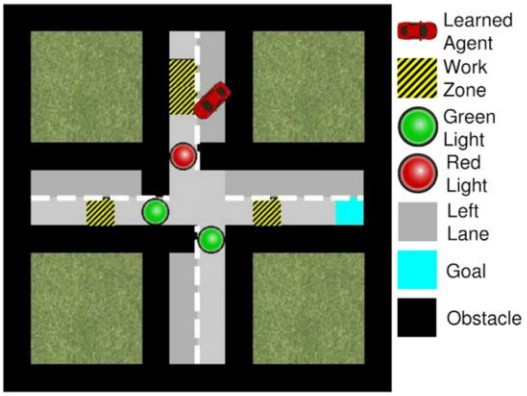
99.7% performance on
1000 rollouts
(No additional training)

20

I should note that one limitation of these manipulations is that you can only use the existing states and propositions to manipulate the policy. The way it is now you cannot add new states or propositions.

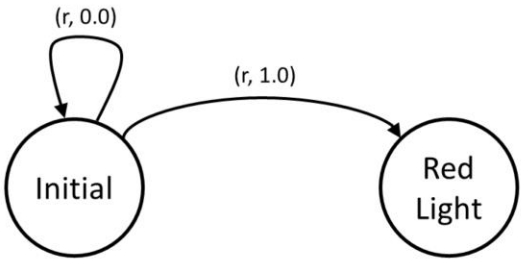
Case Study: Driving

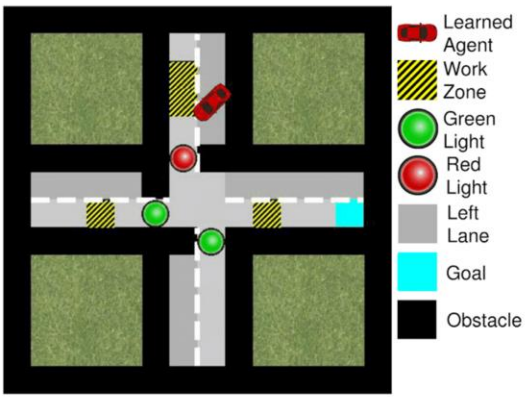




21

Case Study: Driving

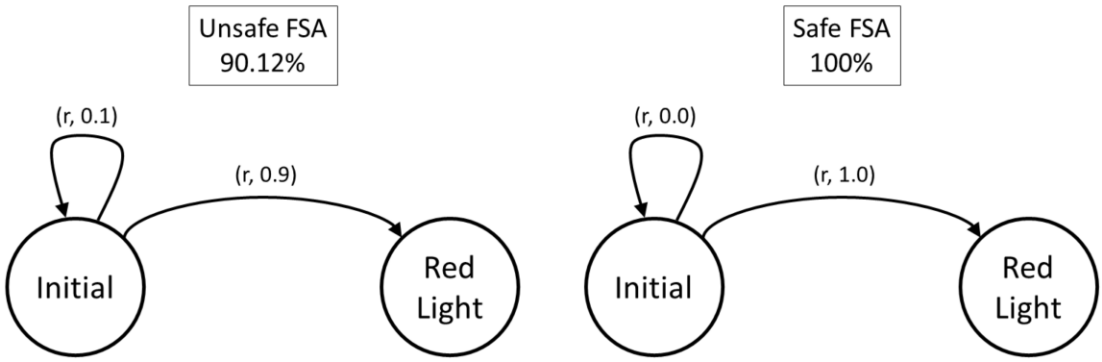




22

Case Study: Driving

- Safe runs on 1000 rollouts:



23

Next Steps

- Assume rules can be encoded as a Finite State Automaton (FSA)
- Assume relevant features of the environment can be detected
- ~~Assume number of FSA states is known~~
- ~~Assume environment outputs current FSA state~~
- Infer the number of FSA states and the transitions between them

24

Learning to Plan with Logical Automata

Brandon Araki, Kiran Vodrahalli, Cristian-Ioan Vasile, Daniela Rus
araki@mit.edu