Advances in Robotic Learning Paper Summary: Learning to Plan with Logical Automata

Brandon Araki[1,*], Kiran Vodrahalli [2,*], Thomas Leech[1,3], Cristian-Ioan Vasile[1], Mark Donahue[3], Daniela Rus [1]
Presented by: Frank Liu, Evan Lam, Michael Drolet

***Abstract*— This electronic document is a live template. The various components of your paper [title, text, heads, etc.] are already defined on the style sheet, as illustrated by the portions given in this document.**

I. INTRODUCTION

Imagine learning a skill like driving. In order to drive properly you not only need to learn how to drive a car, but also learn the rules of the road. To learn how to drive a car, many of us practiced driving and learning all the mechanics to make the car move. Most of us went to a driving school, where a driving instructor taught us certain driving rules in the United States. Others might watch instructional videos from experts online. One way or another, we developed a mental model of the rules of the road through imitating an expert.

Now there are two parts to this learning. The first part is learning the lower level actions in order to operate and drive a car. The second part is developing a mental model or representation of an interpretable policy, such as the rules of the road. The structure of the learned policy should be grounded in meaningful interpretations.

When learning the rules of the road, a naive assumption is that all experts have taught properly and that all the instruction received is correct. If there are bad or even illegal driving habits, these will need to be corrected to ensure safe driving. In real life if a person runs a red light or makes illegal u-turns, after a certain point a police officer would come and help correct that behavior (through a ticket or more serious consequences). We are able to be corrected because the rules in our heads are manipulable, where a human operator can easily modify a learned policy to perform similar but different policies.

Applying this to robotic learning, the authors work towards teaching a robot to learn from demonstrations not just a low-level policy, but also a high level policy that is interpretable and manipulable. The authors create a Logic-based Value Network (LVIN) which utilize these two principles in learning policies. The policies that a robot learns should be interpretable, where there is a set of learned representation of rules. The behavior of the robot should be manipulable, where the rules can be changed in a predictable way which results in changed behavior. The LVIN model is a recurrent, convolutional neural network which uses value iteration over a learned Markov Decision Process (MDP). This MDP factors into two seperate parts, the first as a finite state automaton (FSA) corresponding to the low-level policy, and a bigger MDP corresponding to the rules in an environment.

A big benefit to this approach to learning is that a robot won't just learn from demonstrations, but can modify the learned policy to be safe. Going back to the driving example, but this time with a robot, if a robot was learning the rules of the road and five percent of the training data included say illegal left turns which results in crashes then the robot would learn the policy which crashes five percent of the time. With the author's approach in robotic learning, such a policy can be corrected to stop the crashes. These rules can also be applied in many alternative scenarios.

In this paper, the authors main contributions are

1) A Logic-based Value Network (LVIN) model which learns policies for robotic learning with an imitation learning goal. The authors show the effectiveness of the LVIN model through four different benchmark scenarios.
2) The authors show that the model can learn the transitions from state to state, showing that it can interpret the rules.
3) The authors show that the learning is manipulable, thus generalizing to other tasks and fix mistakes without extra training or experts.

II. RELATED WORK

A. Logic-based Approaches

- Logical structure LTL

B. Multitask and Meta Learning

- Learning many things at the same time.
- One shot learning.

C. Faulty Experts

- Reinforcement learning through imperfect demonstrations
- Intention learning vs Imitation learning

D. Hierarchical Learning

- Reinforcement learning through hierarchy of machines.
- Hierarchical Imitation and Reinforcement learning

[1]MIT CSAIL, Cambridge, MA 02139
[2]Columbia University, New York City, NY 10027
[2]MIT Lincoln Laboratory, Lexington, MA 02421
*Authors contributed equally

III. Problem Statement

The overall goal is of this paper is to create a model that learns from demonstration not just a low-level policy but also a high level policy that is interpretable and manipulable. For our problem statement, we make a few assumptions.

- We assume that rules can be encoded as finite state automaton (FSA).
- We assume that the relative features in an environment can be detected.
- We assume that the FSA states are known.
- We assume that the environment outputs current FSA state as well as low level state at each time step.
- The learned policy comes from the learned transitions among the FSA states and low-level transitions.

When authors say that they can detect the relative features of an environment, what they mean is that these features can be treated as logical propositions or a true/false variable. For example in a 2D grid environment, if there is a red light in the environment then at that particular grid position (x,y) where the red light is located the variable would be set to true. If there was no red light, then the variable would be set to false.

We assume that the expert is following some sort of finite state controller. This finite state maps to the variables in the environment with different states corresponding to the variables environmental. There are different actions based on the variable. For example, move forward on a green light as an action in the state machine. The FSA which the expert follows is the overall policy in which our network wants to learn. The learnt FSA is interpretable because the model can learn expert trajectories. Additionally, this model can obtain new policies without re-learning a changed expert FSA.

A. Value Iteration Network

The LVIN network is based off of the Value Iteration Network [2]. The Value Iteration Network architecture is a fully differentiable version of value iteration. The Q function is calculated using a convolution and the Value function is calculated using a max pool function. This lets you learn the reward function and transition function. One limitation is that this Value Iteration Network is that it works best in a 2D grid environment due to the structure of the network.

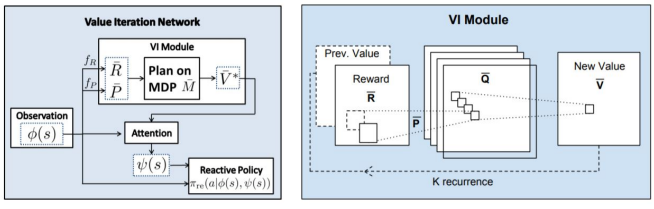

Fig. 1. A block diagram of the Value Iteration Network. The module is shown on the right. The reward function is R, while the transition function is P.

B. Logic-Based Value Iteration Network

There are two differences between the Logic-Based Value Iteration Network and the Value Iteration Network.

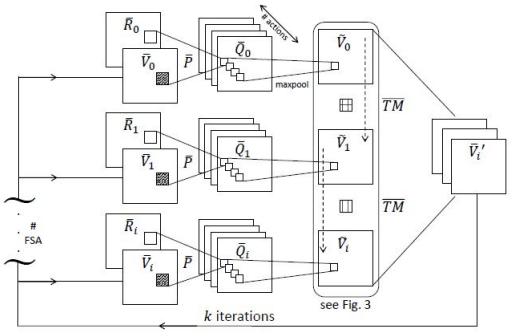

Fig. 2. A block diagram of the Logic-Based Value Iteration Network, figure 4 in the paper [1].

The first difference is that LVIN has a Value Iteration Network for every FSA state. As a result, the reward function and transition function are learned for every FSA state.

The second difference is there is a second convolution which is applied afterwards, which is represented by TM. This convolution is based off the transitions of the FSA. The second convolution helps the model learn the transitions of the FSA's.

IV. Experiments

Use this sample document as your LaTeX source file to create your document. Save this file as **root.tex**. You have to make sure to use the cls file that came with this distribution. If you use a different style file, you cannot expect to get required margins. Note also that when you are creating your out PDF file, the source file is only part of the equation. *Your TEX → PDF filter determines the output file size. Even if you make all the specifications to output a letter file in the source - if your filter is set to produce A4, you will only get A4 output.*

It is impossible to account for all possible situation, one would encounter using TEX. If you are using multiple TEX files you must make sure that the "MAIN" source file is called root.tex - this is particularly important if your conference is using PaperPlaza's built in TEX to PDF conversion tool.

A. Headings, etc

Text heads organize the topics on a relational, hierarchical basis. For example, the paper title is the primary text head because all subsequent material relates and elaborates on this one topic.

B. Figures and Tables

Positioning Figures and Tables: Place figures and tables at the top and bottom of columns. Avoid placing them in the middle of columns. Large figures and tables may span across both columns. Figure captions should be below the figures;

table heads should appear above the tables. Insert figures and tables after they are cited in the text.

TABLE I
AN EXAMPLE OF A TABLE

One	Two
Three	Four

We suggest that you use a text box to insert a graphic (which is ideally a 300 dpi TIFF or EPS file, with all fonts embedded) because, in an document, this method is somewhat more stable than directly inserting a picture.

Fig. 3. Inductance of oscillation winding on amorphous magnetic core versus DC bias magnetic field

Figure Labels: Use 8 point Times New Roman for Figure labels.

V. DISCUSSION AND ANALYSIS

The authors mentioned was that the LVIN model works for finite grid world environments. These finite grid world environments are limited to a 2D grid. These applications might be incredibly useful integrated into a factory workplace with robots in highly controlled environments.

However, the real world is three dimensional and highly unpredictable. These finite grid world environment limits LVIN's real world use case. While the authors have a real world experiment with the Jaco arm to show the LVIN model's effectiveness, the real world experiment exists in a highly controlled environment which is not representative of real world situations.

A logical next step could be extending the 2D grid into a 3D grid world. We would imagine that the larger MDP for the rules of the world would add another dimension and the smaller FSA would add more states. Overall the new network model would be more complex. While the complexity of the network would increase, we do not believe the complexity change would not be too big of a problem in training on modern computers which train significantly large machine learning models.

It also seems the manipulability of the learning is limited. The authors modify the state after detecting an error and to correct bad behavior. It would be quite interesting to see if the model can automatically detect errors and self correct. We imagine there can be neural network infrastructure which can be added for error detection.

It would be quite interesting to explore the LVIN model with moving objects/obstacles in the 2D grid environment. How much would that effect the learned policies and how much impact would introducing such dynamic objects change the output behavior.

VI. CONCLUSIONS

In conclusion, the authors introduce a Logic-Based Value Iteration network which can learn policies from imitation learning and demonstration. The authors tackle how to generalize a learned policy for a particular behavior to a larger set of tasks. Additionally, the authors address how to deal with incorrectly learned policies from incorrect demonstration. This network is a combination of a finite state automaton and a larger markov decision process. The LVIN network is a generalization of the Value Iteration Network, where the LVIN network learns the relevant transitions and creates a policy from the transitions of the FSA's. The key idea of the LVIN network is that a value iteration module is added to the end of a FSA and these modules get linked together.

The authors measure the LVIN network performance in four different virtual domains (Kitchen, Longterm, Pick-world, and Driving) and a real world implementation with a jaco arm. The LVIN network has 99.84 percent, 100 percent, 83.2 percent, and 99.6 percent, 89.8 percent success rates respectively. The model is shown to be accurate and effective in generalizing to new task specifications, and correcting errors. Future works can focus on expanding the model dimensionality for 3D scenarios and even self learn errors and dynamic objects in the world.

REFERENCES

[1] Araki, Brandon & Vodrahalli, Kiran & Leech, Thomas & Vasile, Cristian-Ioan & Donahue, Mark & Rus, Daniela. (2019). Learning to Plan with Logical Automata.

[2] Aviv Tamar, Yi Wu, Garrett Thomas, Sergey Levine, and Pieter Abbeel. Value iteration networks. In Advances in Neural Information Processing Systems 29, pages 2154– 2162, 2016.