|  |  |  |  |
| --- | --- | --- | --- |
| **Title** | NIPS2016 | Value Iteration Networks: Aviv Tamar, UC Berkeley | |
| **Abstract** | > They introduce the value iteration network (VIN): a fully differentiable neural network with a ‘planning module’ embedded within.  > The VIN can be used to predict outcomes that involve planning-based reasoning, such as policies for reinforcement learning.  > Key to the VIN is a novel differentiable approximation of the value-iteration algorithm, represented by a CNN, and trained using standard backpropagation.  > Evaluate the VIN based policies on discrete and continuous path-planning domains, and on a natural-language based search task. | | |
| **Introduction** | ***Methods*** | Pros | Cons |
| ***Classic deepRL*** | > Neural network is trained to represent a policy: mapping <observation> to <action>, the goal is get a policy has good long-term behavior. | > Supervised learning is one-step decisions making. However, RL needs some form of planning.  > Recent deepRL model are inherently *reactive*, and lack explicit planning computation. The success of *reactive* policy in sequential problems is duo to the *learning algorithm,* the algorithm is trained to select actions that have good-long term consequences in its training domain. |
| ***Model RL*** |  | Require system identification, infeasible for complex real system. |
| ***The cons example of Recent deepRL (reactive strategy)*** | People's expectations of such type of games are after training a policy to solve several instances of this problem with different obstacle configuration, the policy would generalize to solve a different, unseen domain configuration. However, in experiment, with standard CNN-based network, we can easily solve a set of such maps, however, the policy trained do not generalize to new tasks outside this set, because they do not understand the goal-directed nature of behavior. | |
| ***Proposed Method*** | > They proposed a NN-based policy that can effectively learn to plan. Their model, termed a value-iteration network (VIN), has a differentiable 'planning program' embedded within in the NN structure.  > VIN could be trained model free, without requiring explicit system identification. Moreover, the errors in VINs can be mitigated by training the network end-to-end. | |
| **Background** | ***Value Iteration*** | The goal in an MDP is to find a policy that obtains high rewards in the long term | |
| ***Convolutional Neural Networks(CNN)*** |  | |
| ***Reinforcement Learning and Imitation Learning*** | > When MDP transitions or rewards are not know in advance, planning algorithms cannot be applied. In this cases, a policy can be learned from either *expert supervision -IL or by trail and error -RL.*  *>* In imitation learning, learning a policy becomes an instance of supervised learning  > In reinforcement learning, the agent can act and observe the rewars and the state transitions its action effect. | |
| The Value Iteration Network Model | Overview of the Model | Notation  : denotes the MDP of the domain for which we design our policy.  : unknown, contains the useful information about the optimal policy in the original task.  Are the states, actions, and rewards and transitions in  are depends on the observation in M.  is the parameters of  > This section mainly focus on how to use the planning result with in the NN policy . The approach is based on two important observation. The first is that the vector of encodes all the information about the optimal plan in . The second is that the MDP has a local connectivity structure, it only depends on a subset of the values of ，which refers to an attention module. | |
| The VI Module | > Each channel in the convolution layer corresponds to the Q-function for a specific action.  > Each convolution kernel weights correspond to the discounted transition probabilities.  > Representing VI in this form makes learning the MDP parameters and reward function natural- by back propagating through the network. | |
| Value Iteration Networks | > The VIN is based on the general planning-based policy defined above, with the VI module as the planning algorithm.  > In order to implement a VIN, one has to specify the state and action spaces for the planning module | |
| Experiments | Overview | The goal in these experiments:  > Can VIN effectively learn a planning computation using standard RL and IL algorithms?  > Does the planning computation learned by VINs make them better than reactive policies at generalizing to new domains? | |
| Grid-World Domain |  | |
| Mars Rover Navigation |  | |
| Continuous Control |  | |
| Conclusions |  |  | |