VALUE ITERATION NETWORKS

Aviv Tamar
Joint work with Pieter Abbeel, Sergey Levine, Garrett Thomas, Yi Wu

June 23, 2016

UC Berkeley

Berkeley
Artificial Intelligence Research Laboratory

INTRODUCTION

MOTIVATION

- Goal: autonomous robots

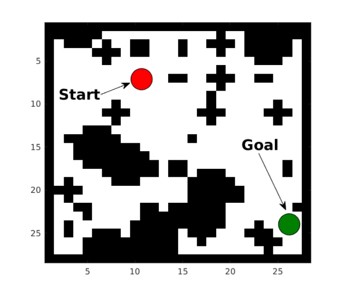
Robot, bring me the milk bottle!

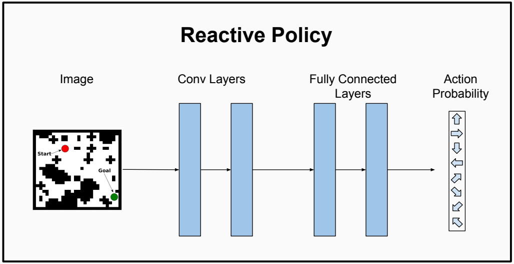
http://www.wellandgood.com/wp-content/uploads/2015/02/Shira-fridge.jpg

- Solution: RL?

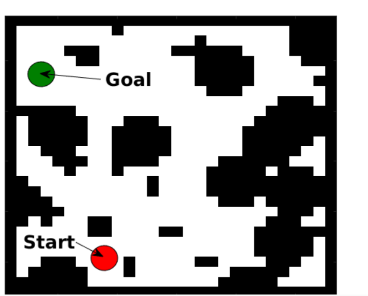
- Deep RL learns policies from high-dimensional visual input[1,2]
- Learns to act, but does it **understand**?
- A simple test: generalization on grid worlds

[1] Mnih et al. Nature 2015
[2] Levine et al. JMLR 2016

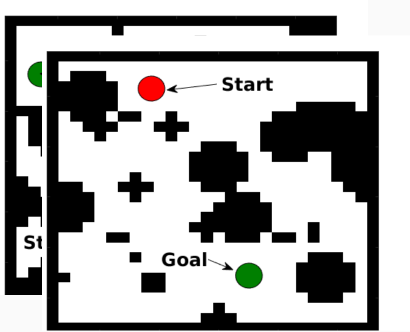

Start
Goal
5
10
15
20
25
5
10
15
20
25


Reactive Policy
Image
Conv Layers
Fully Connected Layers
Action Probability
Start
Goal

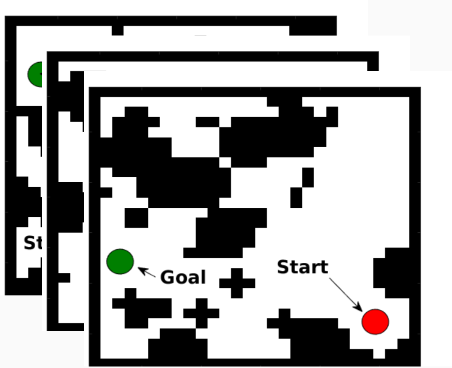
Train

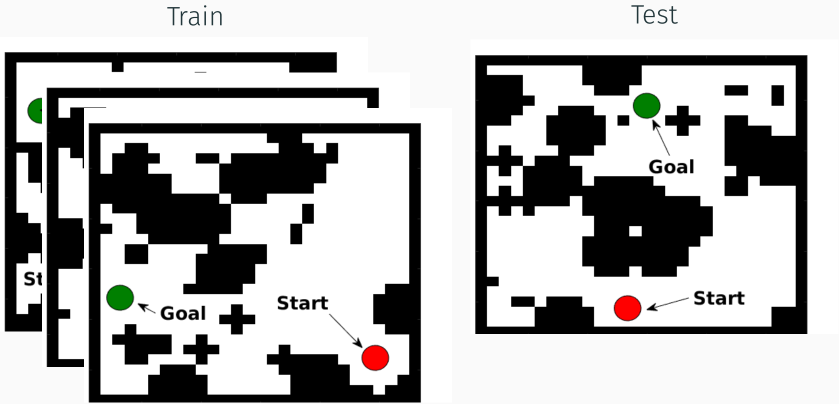


Train



Train





Observation: reactive policies do not generalize well

Why don't reactive policies generalize?

- A sequential task requires a planning computation
- RL gets around that – learns a mapping
  - State $\rightarrow$ Q-value
  - State $\rightarrow$ action with high return
  - State $\rightarrow$ action with high advantage
  - State $\rightarrow$ expert action
  - [State] $\rightarrow$ [planning-based term]
- Q/return/advantage: planning **on training domains**
- New task – need to re-plan

In this work:

- Learn to plan
- Policies that generalize to unseen tasks

BACKGROUND

Planning in MDPs

- States $s \in \mathcal{S}$, actions $a \in \mathcal{A}$
- Reward $R(s, a)$
- Transitions $P(s'|s, a)$
- Policy $\pi(a|s)$
- Value function $V^\pi(s) \doteq \mathbb{E}^\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \middle| s_0 = s \right]$
- Value iteration (VI)

$$V_{n+1}(s) = \max_a Q_n(s, a) \quad \forall s,$$
$$Q_n(s, a) = R(s, a) + \gamma \sum_{s'} P(s'|s, a) V_n(s').$$

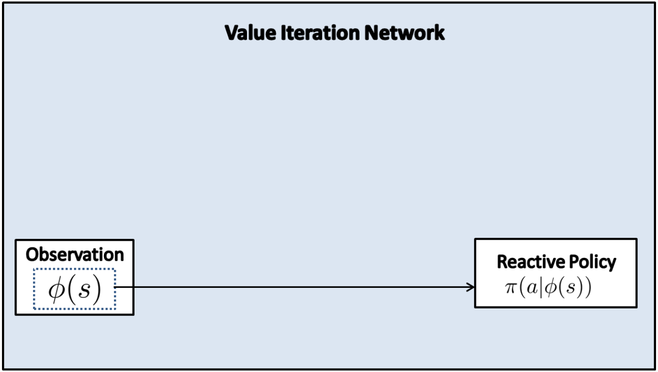
- Converges to $V^* = \max_\pi V^\pi$
- Optimal policy $\pi^*(a|s) = \arg\max_a Q^*(s, a)$

Policies in RL / imitation learning

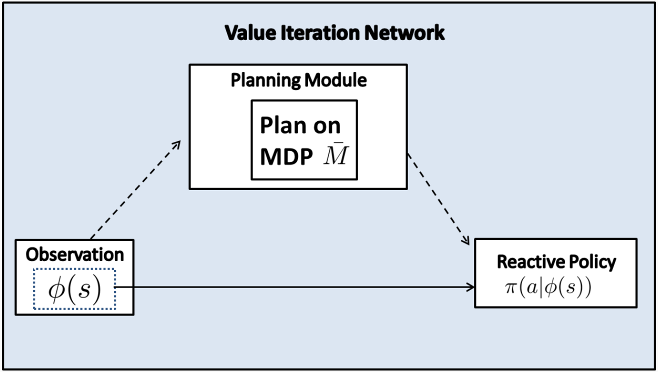
- State observation $\phi(s)$
- Policy: $\pi_\theta(a|\phi(s))$
  - Neural network
  - Greedy w.r.t. Q (DQN)
- Algorithms perform SGD, require $\nabla_\theta \pi_\theta(a|\phi(s))$
- Only loss function varies
  - Q-learning (DQN)
  - Trust region policy optimization (TRPO)
  - Guided policy search (GPS)
  - Imitation Learning (supervised learning, DAgger)
- Focus on policy representation
- Applies to model-free RL / imitation learning

A MODEL FOR POLICIES THAT PLAN

- Start from a reactive policy

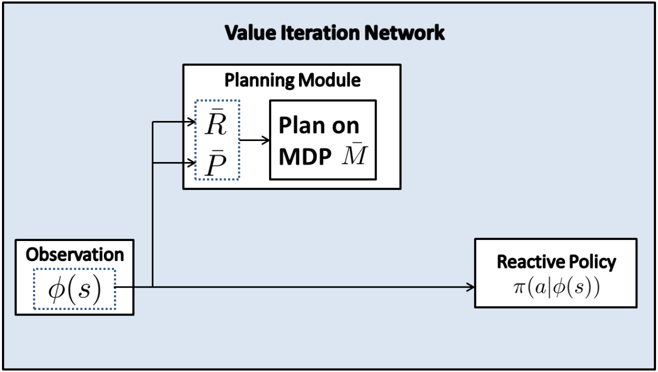


- Add an explicit planning computation
- Map observation to planning MDP $\bar{M}$



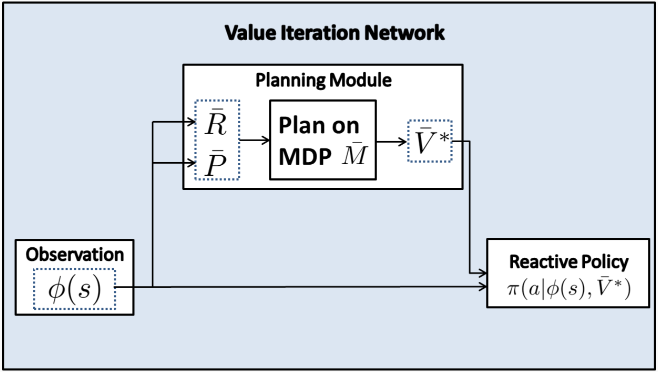
- Assumption: observation can be mapped to a useful (but **unknown**) planning computation

- NNs map observation to reward and transitions
- Later - learn these



How to use the planning computation?

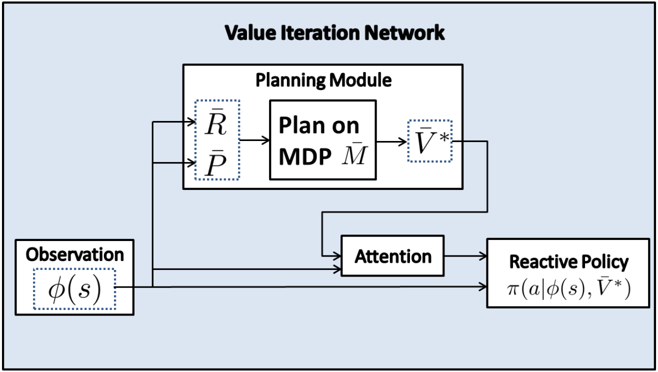
- Fact 1: value function = sufficient information about plan
- Idea 1: add as features vector to reactive policy



- Fact 2: action prediction can require only subset of $\bar{V}^*$

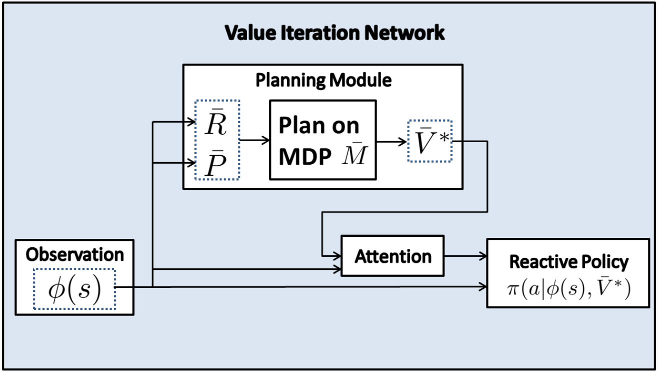
$$\pi^*(a|s) = \arg\max_a R(s,a) + \gamma \sum_{s'} P(s'|s,a) V^*(s')$$

- Similar to attention models, effective for learning[1]



[1] Xu et al. ICML 2015

- Policy is still a mapping $\phi(s) \rightarrow \text{Prob}(a)$
- Parameters θ for mappings $\bar{R}$, $\bar{P}$, attention
- Can we backprop?



How to backprop through planning computation?

VALUE ITERATION = CONVNET

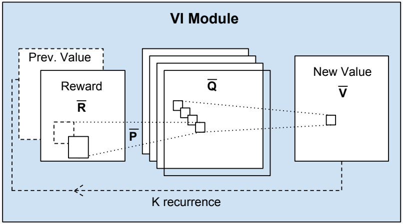
Value iteration

K iterations of:

$$\bar{Q}_n(\bar{s},\bar{a}) = \bar{R}(\bar{s},\bar{a}) + \sum_{\bar{s}'} \gamma \bar{P}(\bar{s}'|\bar{s},\bar{a}) \bar{V}_n(\bar{s}')$$

$$\bar{V}_{n+1}(\bar{s}) = \max_{\bar{a}} \bar{Q}_n(\bar{s},\bar{a}) \quad \forall \bar{s}$$

Convnet

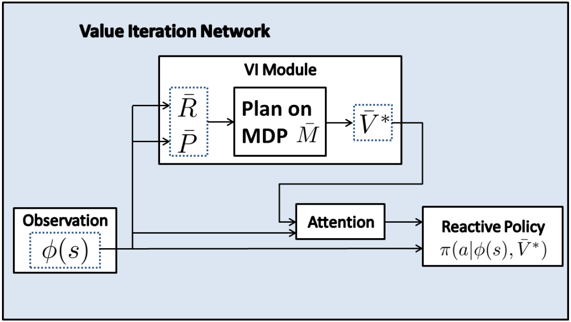


- $\bar{\mathcal{A}}$ channels in $\bar{Q}$ layer
- Linear filters $\Longleftrightarrow \gamma\bar{P}$
- Tied weights
- Channel-wise max-pooling

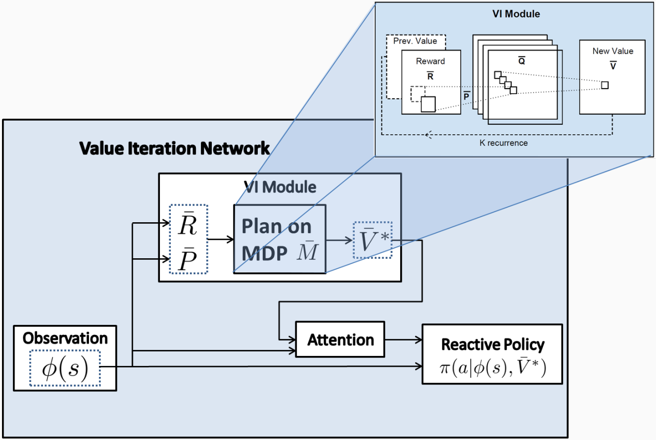
- Best for locally connected dynamics (grids, graphs)
- Extension – input-dependent filters

VALUE ITERATION NETWORKS

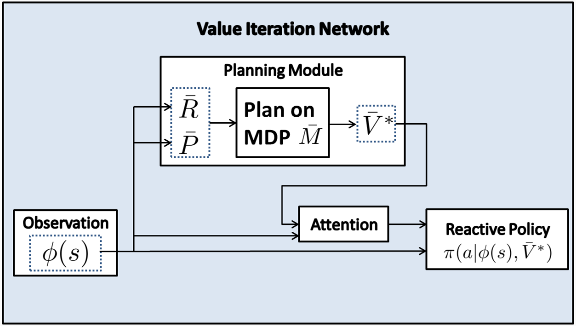
- Use VI module for planning



- Value iteration network (VIN)



- Just another policy representation $\pi_\theta(a|\phi(s))$
- That can learn to plan
- Train like any other policy!
- Backprop – just like a convnet
- Implementation – few lines of Theano code



EXPERIMENTS

Questions

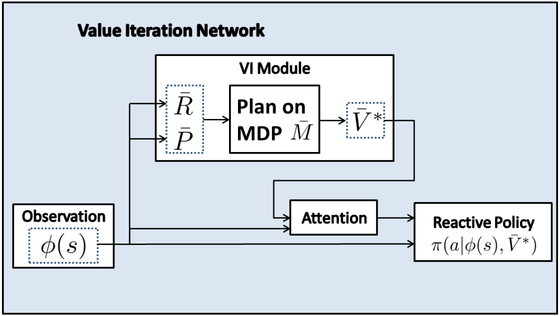
1. Can VINs learn a planning computation?
2. Do VINs generalize better than reactive policies?

GRID-WORLD DOMAIN

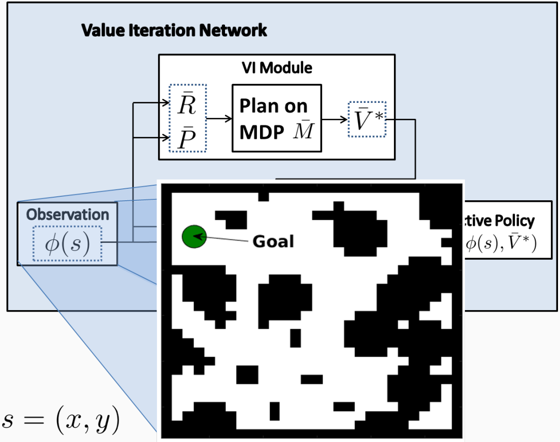
GRID-WORLD DOMAIN

- Supervised learning from expert (shortest path)
- Observation: image of obstacles + goal, current state
- Compare VINs with reactive policies

- VI state space: grid-world
- VI Reward map: convnet
- VI Transitions: 3×3 kernel
- Attention: choose $\bar{Q}$ values for current state
- Reactive policy: FC, softmax

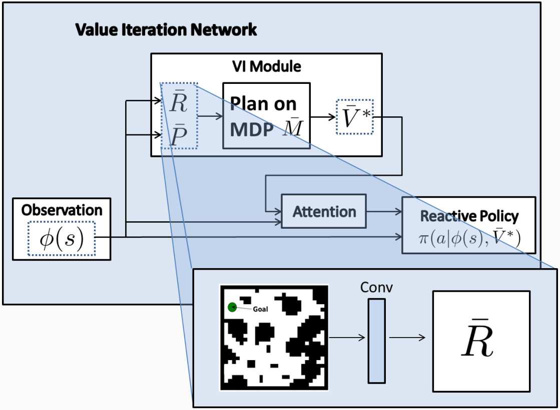


- VI state space: grid-world
- VI Reward map: convnet
- VI Transitions: 3×3 kernel
- Attention: choose $\bar{Q}$ values for current state
- Reactive policy: FC, softmax



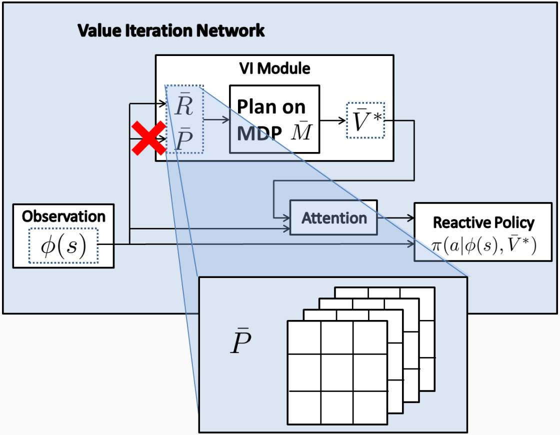
GRID-WORLD DOMAIN

- VI state space: grid-world
- VI Reward map: convnet
- VI Transitions: 3×3 kernel
- Attention: choose $\bar{Q}$ values for current state
- Reactive policy: FC, softmax



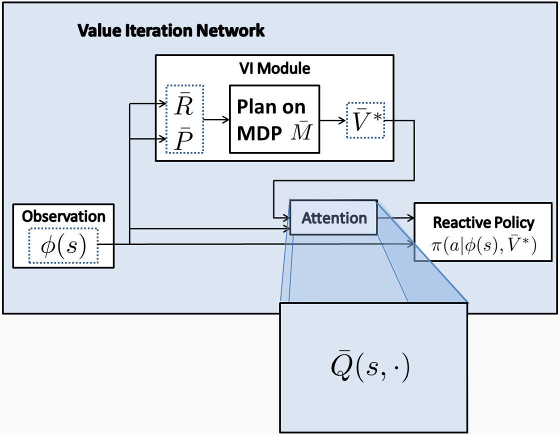
GRID-WORLD DOMAIN

- VI state space: grid-world
- VI Reward map: convnet
- VI Transitions: 3×3 kernel
- Attention: choose $\bar{Q}$ values for current state
- Reactive policy: FC, softmax

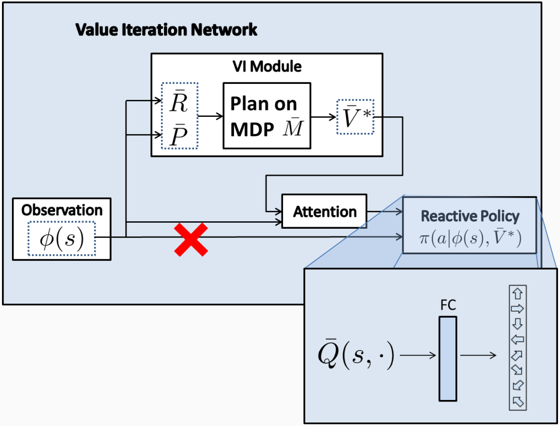


GRID-WORLD DOMAIN

- VI state space: grid-world
- VI Reward map: convnet
- VI Transitions: 3×3 kernel
- Attention: choose $\bar{Q}$ values for current state
- Reactive policy: FC, softmax



- VI state space: grid-world
- VI Reward map: convnet
- VI Transitions: 3×3 kernel
- Attention: choose $\bar{Q}$ values for current state
- Reactive policy: FC, softmax



Compare with:

- CNN inspired by DQN architecture[1]
    - 5 layers
    - Current state as additional input channel
- Fully convolutional net (FCN)[2]
    - Pixel-wise semantic segmentation (labels=actions)
    - Similar to our attention mechanism
    - 3 layers
    - Full-sized kernel – receptive field always includes goal

Training:

- 5000 random maps, 7 trajectories in each
- Supervised learning from shortest path

[1]Mnih et al. Nature 2015
[2]Long et al. CVPR 2015

Evaluation:

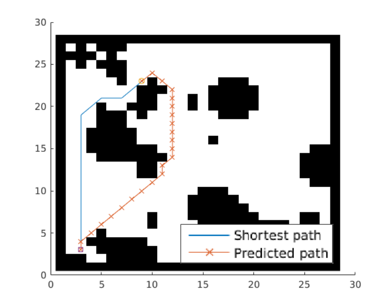
- Action prediction error (on test set)
- Success rate – reach target without hitting obstacles

Results:

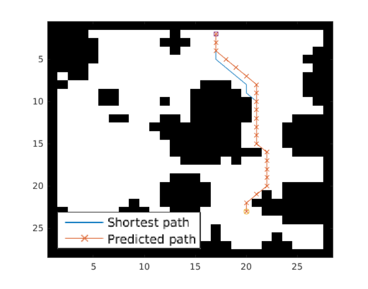
Domain	VIN		CNN		FCN	
	Prediction loss	Success rate	Pred. loss	Succ. rate	Pred. loss	Succ. rate
8×8	0.004	**99.6%**	0.02	97.9%	0.01	97.3%
16×16	0.05	**99.3%**	0.10	87.6%	0.07	88.3%
28×28	0.11	**97%**	0.13	74.2%	0.09	76.6%

VINs learn to plan!

Results:

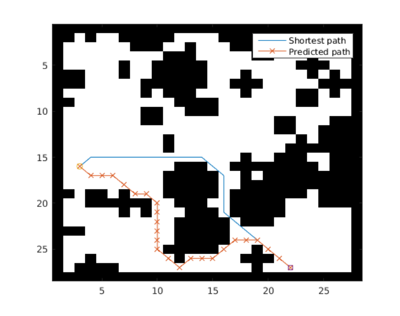


Results:

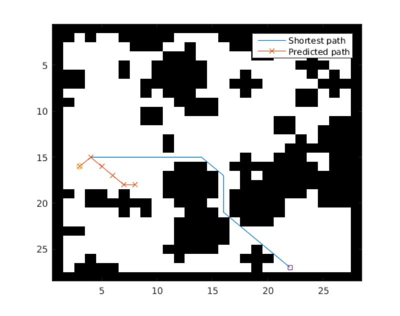


Results:

VIN

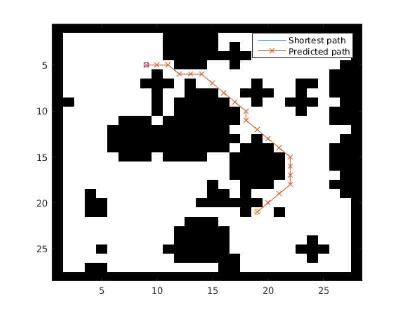


FCN

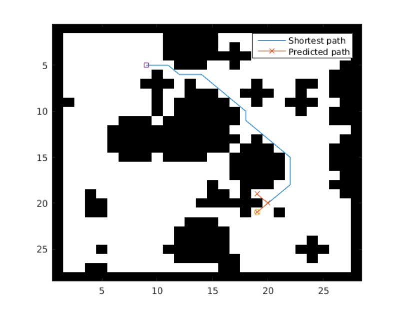


Results:

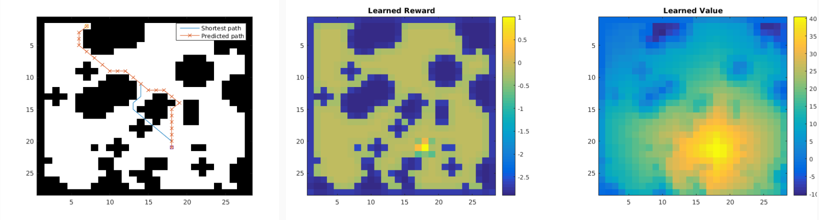
VIN



FCN



Results:

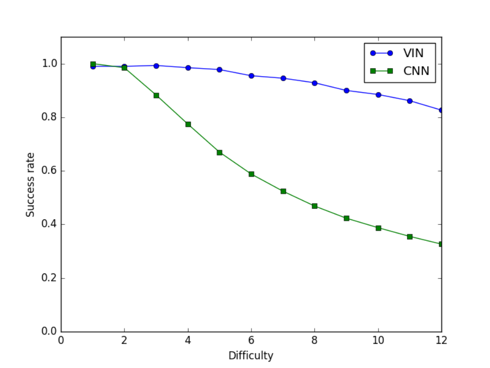

Shortest path
Predicted path
Learned Reward
Learned Value

Depth vs. Planning

- Planning requires **depth** – why not just add more layers?
- Experiment: untie weights in VINs
    - Degrades performance
    - Especially with less data
- The VI structure is important

Training using RL

- Q-learning, TRPO[1]
- Same network structure
- Curriculum learning for exploration
- Similar findings as supervised case

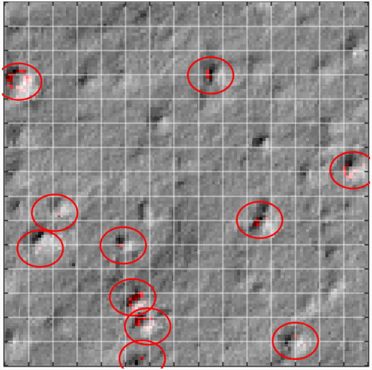


[1] Schulman et al. ICML 2015

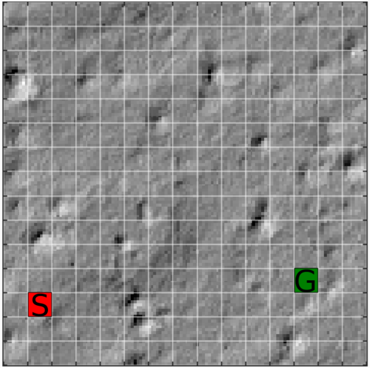
MARS-NAVIGATION DOMAIN

MARS-NAVIGATION DOMAIN

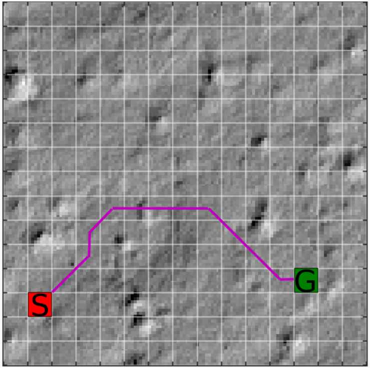
- Grid-world with natural image observations
- Overhead images of Mars terrain
- Obstacle = slope of 10° or more
- Elevation data not part of input



- Grid-world with natural image observations
- Overhead images of Mars terrain
- Obstacle = slope of 10° or more
- Elevation data not part of input

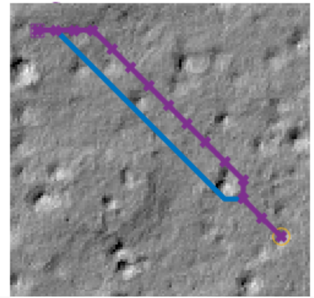


- Grid-world with natural image observations
- Overhead images of Mars terrain
- Obstacle = slope of 10° or more
- Elevation data not part of input



Same grid-world VIN, 3 layers in $\bar{R}$ convnet

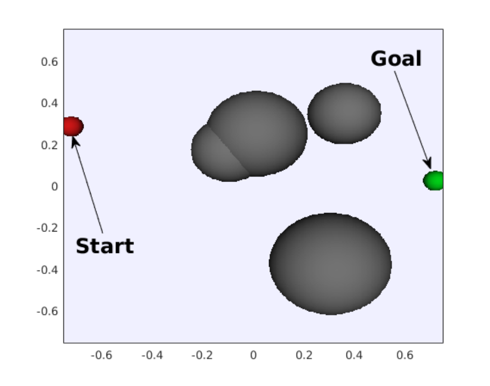
	Pred. loss	Succ. rate
VIN	0.089	84.8%
Best achievable	-	90.3%



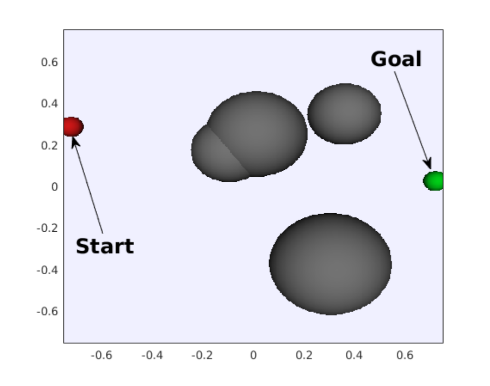
- Best achievable: train classifier with **obstacle labels**, predict map and plan
- VIN **did not** observe any labeled obstacle data
- Conclusion: can handle non-trivial perception

CONTINUOUS CONTROL DOMAIN

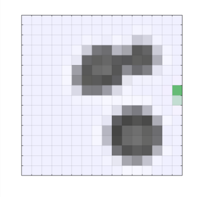
- Move particle between obstacles, stop at goal
- 4d state (position, velocity), 2d action (force)
- Input: state + low-res (16×16) map



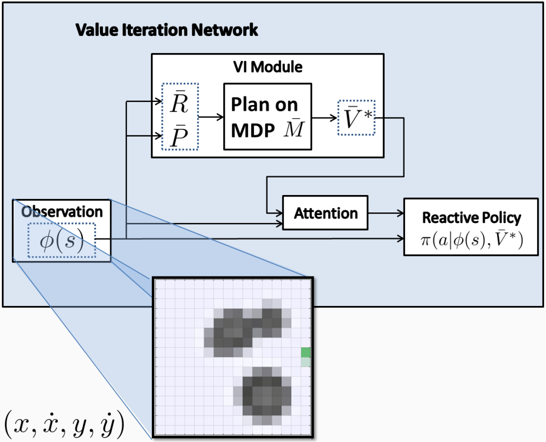
- Move particle between obstacles, stop at goal
- 4d state (position, velocity), 2d action (force)
- Input: state + low-res (16×16) map



Input map

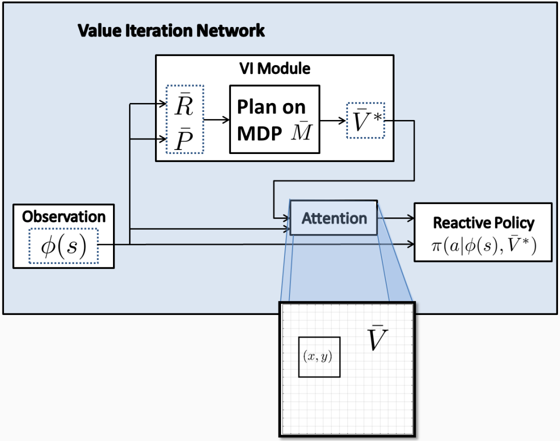


- VI state space: grid-world
- Attention: 5 × 5 patch around current state
- Reactive policy: FC, Gaussian mean output

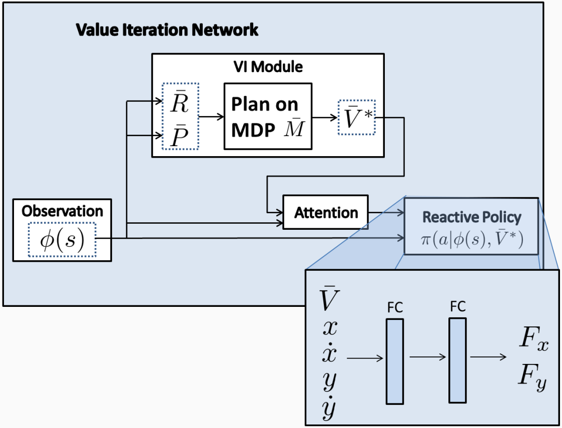


$s = (x, \dot{x}, y, \dot{y})$

- VI state space: grid-world
- Attention: 5×5 patch around current state
- Reactive policy: FC, Gaussian mean output



- VI state space: grid-world
- Attention: 5×5 patch around current state
- Reactive policy: FC, Gaussian mean output



Compare with:

- CNN inspired by DQN architecture[1,2]
  - 2 conv layers + 2×2 pooling + 3 FC layers

Training:

- 200 random maps
- iLQG with unknown dynamics[3]
- Supervised learning (equiv. 1 iteration of guided policy search)

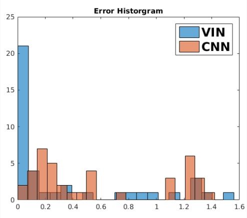
[1] Mnih et al. Nature 2015
[2] Lillicrap et al. ICLR 2016
[3] Levine & Abbeel, NIPS 2014

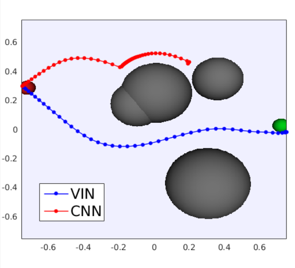
Evaluation:

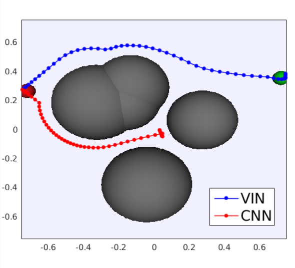
- Distance to goal on final time

Results:

Network	Train Error	Test Error
VIN	0.30	0.35
CNN	0.39	0.58



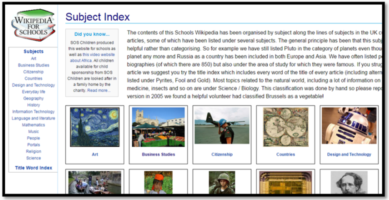




WEB-NAV DOMAIN – LANGUAGE-BASED SEARCH

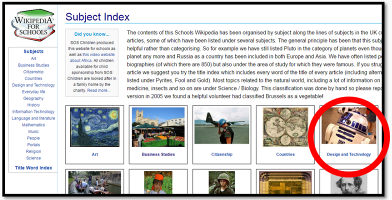
- ”End-to-End Goal-Driven Web Navigation” Nogueira & Cho, arXiv 2016
- Navigate website links to find a query

The Enigma was used commercially from the early 1920s on, and was also adopted by the military and governmental services of a number of nations—most famously by Nazi Germany before and during World War II.
The mechanical parts act in such a way as to form a varying electrical circuit—the actual encipherment of a letter is performed electrically. When a key is pressed, the circuit is completed; current flows through the various components and ultimately lights one of many different lamps, indicating the output letter.



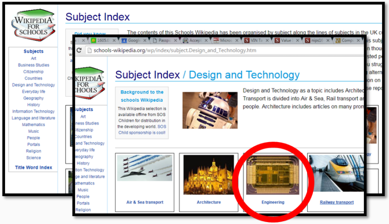
- "End-to-End Goal-Driven Web Navigation" Nogueira & Cho, arXiv 2016
- Navigate website links to find a query

The Enigma was used commercially from the early 1920s on, and was also adopted by the military and governmental services of a number of nations—most famously by Nazi Germany before and during World War II.
The mechanical parts act in such a way as to form a varying electrical circuit—the actual encipherment of a letter is performed electrically. When a key is pressed, the circuit is completed; current flows through the various components and ultimately lights one of many different lamps, indicating the output letter.



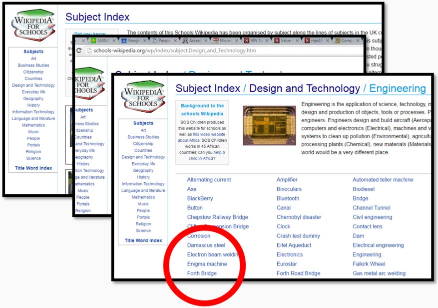
- ”End-to-End Goal-Driven Web Navigation” Nogueira & Cho, arXiv 2016
- Navigate website links to find a query

The Enigma was used commercially from the early 1920s on, and was also adopted by the military and governmental services of a number of nations—most famously by Nazi Germany before and during World War II.
The mechanical parts act in such a way as to form a varying electrical circuit—the actual encipherment of a letter is performed electrically. When a key is pressed, the circuit is completed; current flows through the various components and ultimately lights one of many different lamps, indicating the output letter.



- ”End-to-End Goal-Driven Web Navigation” Nogueira & Cho, arXiv 2016
- Navigate website links to find a query

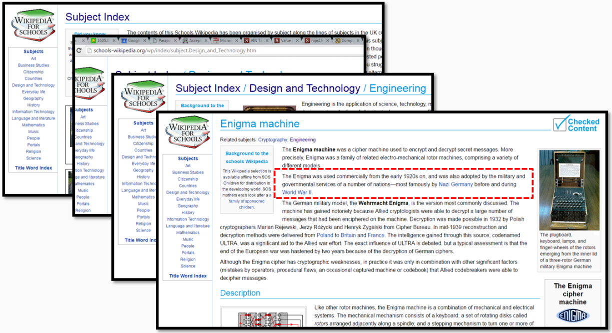
The Enigma was used commercially from the early 1920s on, and was also adopted by the military and governmental services of a number of nations—most famously by Nazi Germany before and during World War II.
The mechanical parts act in such a way as to form a varying electrical circuit—the actual encipherment of a letter is performed electrically. When a key is pressed, the circuit is completed; current flows through the various components and ultimately lights one of many different lamps, indicating the output letter.



WEB-NAV DOMAIN

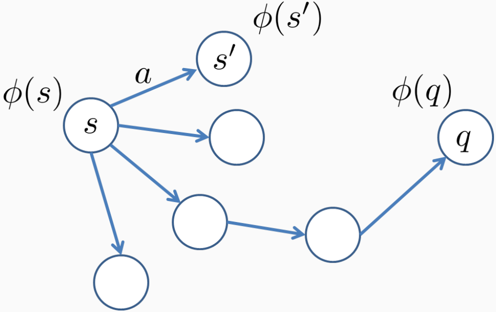
- "End-to-End Goal-Driven Web Navigation" Nogueira & Cho, arXiv 2016
- Navigate website links to find a query

The Enigma was used commercially from the early 1920s on, and was also adopted by the military and governmental services of a number of nations—most famously by Nazi Germany before and during World War II.
The mechanical parts act in such a way as to form a varying electrical circuit—the actual encipherment of a letter is performed electrically. When a key is pressed, the circuit is completed; current flows through the various components and ultimately lights one of many different lamps, indicating the output letter.

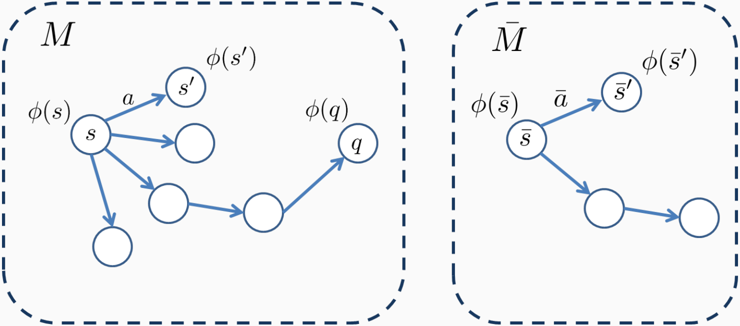


WEB-NAV DOMAIN

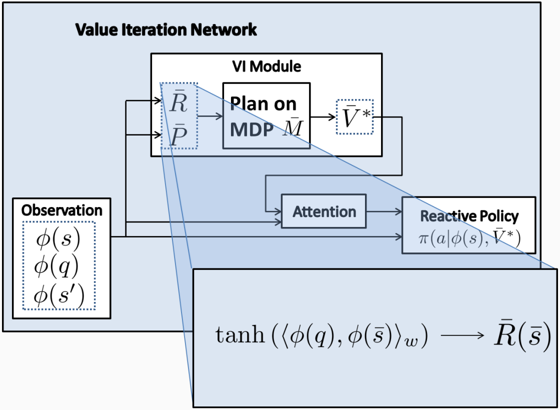
- "End-to-End Goal-Driven Web Navigation" Nogueira & Cho, arXiv 2016
- Navigate website links to find a query
- Observe: $\phi(s)$, $\phi(q)$, $\phi(s'|s,a)$
- Features: average word embeddings
- Baseline policy: $h = \text{NN}(\phi(s), \phi(q))$, $\quad \pi(a|s) \propto \exp(\langle h, \phi(s')\rangle)$



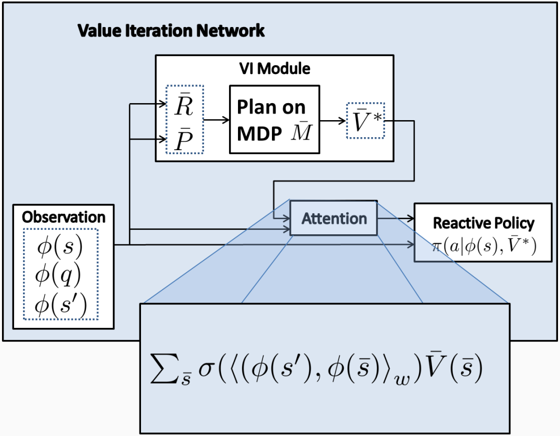
- Idea: use an approximate graph for planning
- Wikipedia for Schools website (6K pages)
- Approximate graph: 1st+2nd level categories (3%)



- VI state space + transitions : approx. graph
- VI Reward map: weighted similarity to ϕ(q)
- Attention: average weighted by similarity to ϕ(s′)
- Reactive policy: add feature to ϕ(s′)



- VI state space + transitions : approx. graph
- VI Reward map: weighted similarity to ϕ(q)
- Attention: average weighted by similarity to ϕ(s′)
- Reactive policy: add feature to ϕ(s′)



Evaluation:

- Success – all correct actions within top-4 predictions
- Test set 1: start from index page

Results:

Network	Success set 1	
Baseline	1025/2000	
VIN	1030/2000	

Evaluation:

- Success – all correct actions within top-4 predictions
- Test set 1: start from index page
- Test set 2: start from random page

Results:

Network	Success set 1	Success set 2
Baseline	1025/2000	304/4000
VIN	1030/2000	346/4000

Evaluation:

- Success – all correct actions within top-4 predictions
- Test set 1: start from index page
- Test set 2: start from random page

Results:

Network	Success set 1	Success set 2
Baseline	1025/2000	304/4000
VIN	1030/2000	346/4000

Preliminary results: full English Wikipedia website, using wiki-school as approximate graph

SUMMARY & OUTLOOK

SUMMARY

- Learn to plan → generalization
- Framework for planning based NN policies
  - Motivated by dynamic programming theory
  - Differentiable planner (VI = CNN)
  - Compositionality of NNs – perception & control
  - Exploits flexible prior knowledge
  - Simple to use

- Different planning algorithms
  - MCTS
  - Optimal control[1]
  - Inverse RL[2]
- How to obtain approximate planning problem
  - Game manual in Atari
- Generalization in RL[3]
  - theory?
  - benchmarks?
  - Algorithms?
- Generalization $\neq$ lifelong RL, transfer learning[4]
- Hierarchical policies, but not options/skills/etc.

[1] Watter et al. NIPS 2015
[2] Zucker & Bagnell, ICRA 2011
[3] Oh et al. ICML 2016, Barreto et al. arXiv 2016
[4] Taylor & Stone, JMLR 2009

THANK YOU!