ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

Flatland Challenge

Deep learning course final project

Alessio Falai, Leonardo Calbi

January 14, 2021

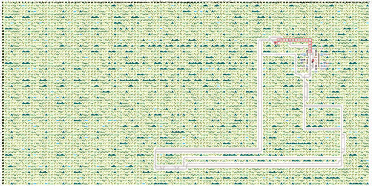
Alma Mater Studiorum - University of Bologna

page

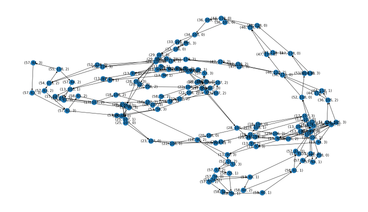
Environment

Railway encoding

- Standard encoding: dense, matrix-based
- Custom encoding: sparse, graph-based
  - Only stores meaningful positions: junctions and targets
  - Memory efficient



(a) Grid



(b) COJG

Predictions

- Standard predictions: shortest path only, computed on the matrix encoding
- Custom predictions: shortest and deviation paths, computed on the custom graph encoding
  - Scalable solution
  - Incremental computation
  - Time efficient

Improvements

- Choices: reduced action space
  - Follow the track to the left, to the right or stop
- Real decisions: only relevant calls to the policy estimator
  - On before join, at fork or in a combination of the two
- Action masking: only consider legal moves
  - Stop is legal only when there is more than one agent
- Rewards shaping
  - Incremental rewards (related to real decisions)
  - Deadlock penalty
  - Target arrival bonus
  - Consecutive stop action penalty

Architectures

- From state representation to Q-values
- Standard experience replay
- Regression framework: TD error minimization
  - MSE or Huber loss
- Double, dueling version with custom Bellman operators

- From graph nodes to an embedding space
- Two main uses: the input graph could be . . .
  - . . . railway encoding
  - . . . agent communication network
- Different graph convolutional layer
  - GCN as a simple sum/mean neighborhood aggregator
  - GAT as a learnable information gatherer, with attention over features of nearby agents

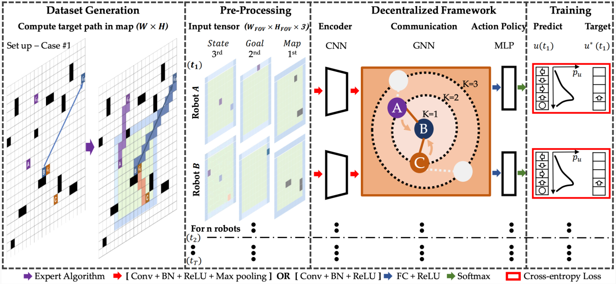
Model flows

Binary tree

- Binary tree observation
  - Shaped like the standard tree observation, with two branches (left and right choice)
  - Based on the graph railway encoding (nodes in the tree are linked to nodes in the graph)
  - Prior knowledge injection: only nodes corresponding to predictions (shortest/deviation) are filled with features
  - Carefully designed features (based on direction, speed and malfunction of agents)
- Per-agent flow
  1. Compute, linearize and normalize the binary tree
  2. The observation is fed to the DQN, which returns Q-values
  3. The action selector chooses an action

- Graph observation
  - Exploit the railway encoding graph and assign simple features to each node (cell status, target distance)
  - Does not make use of predictions
- Per-agent flow
  1. The GNN computes convolutions over the input graph and extracts embeddings from pre-specified positions
  2. Such embeddings are given as input to the DQN, which returns Q-values
  3. The action selector chooses an action

- FOV observation
  - Square Field Of View centered around each agent, of dimension (c, d, d)
  - Channel dimension c used to represent different features
  - Makes use of the shortest/deviation paths prediction

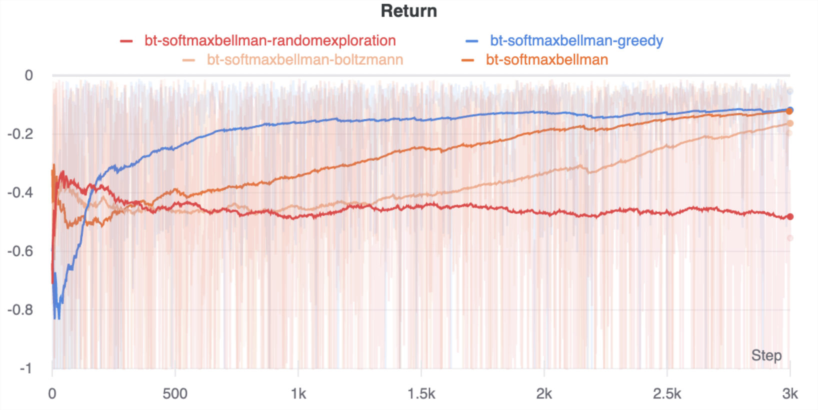


- All agents flow
  1. The FOV observation is computed for all agents at once
  2. A CNN (built by blocks of Conv + BN + ReLU + MaxPool) is used to extract higher-level features from FOVs
  3. The CNN output is compressed into a fixed size by an MLP
  4. A GNN takes as input a graph where nodes are agents (features are MLP outputs) and edges encode their proximities in the environment
  5. The output embeddings of the GNN are used as input to the DQN, which returns Q-values
  6. The action selector chooses an action for each agent

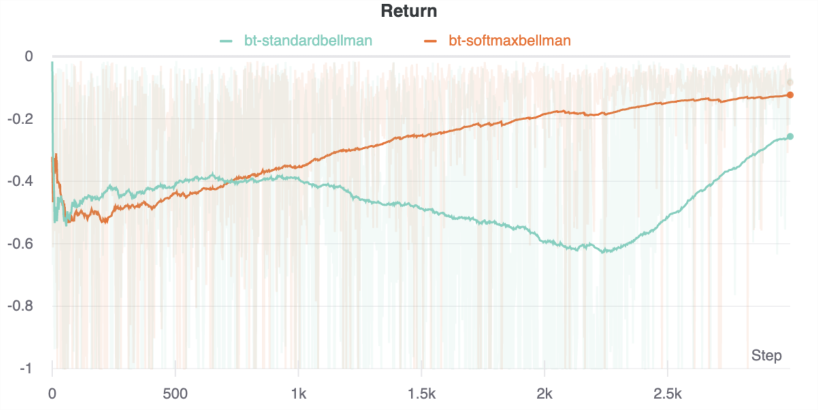
Experiments

Action selectors

- ϵ-greedy vs. Boltzmann
- Random and greedy baselines



- Standard vs. softmax Bellman



Results

Environment

Parameter	Medium	Big
`width`	48	64
`height`	27	36
`max_cities`	5	9
`max_rails_between_cities`	2	5
`max_rails_in_cities`	3	5

Complications

Parameter	A	B
`speeds`	1	$\{1, \frac{1}{2}, \frac{1}{3}, \frac{1}{4}\}$
`malfunctions.rate`	0	$\frac{1}{200}$
`malfunctions.min_duration`	-	15
`malfunctions.max_duration`	-	50

- Model flow
  - Binary tree observator with DQN model
  - ϵ-greedy action selector and softmax Bellman operator

	3 agents		5 agents		7 agents	
	A	B	A	B	A	B
Dones	93.07%	83.80%	89.40%	76.64%	82.51%	67.66%
Deadlocks	3.07%	5.80%	5.96%	13.40%	10.23%	12.54%
Return	-0.1404	-0.1459	-0.1581	-0.1688	-0.2057	-0.1940
Steps	142	339	187	415	250	445

Big environment

- Model flow: the same one used in the medium setting

	5 agents		7 agents		10 agents	
	A	B	A	B	A	B
Dones	86.28%	68.76%	84.17%	61.43%	76.90%	50.28%
Deadlocks	3.24%	12.88%	5.89%	20.97%	12.54%	31.82%
Return	-0.2331	-0.2081	-0.2449	-0.2260	-0.2790	-0.2615
Steps	400	637	447	650	497	675

Thank you for your attention