Politecnico di Milano

Computer Science and Engineering

HONOURS PROGRAMME HP-SR in Information Technology

Neural Weighted A*

Learning Graph Costs and Heuristics with Differentiable Anytime A*

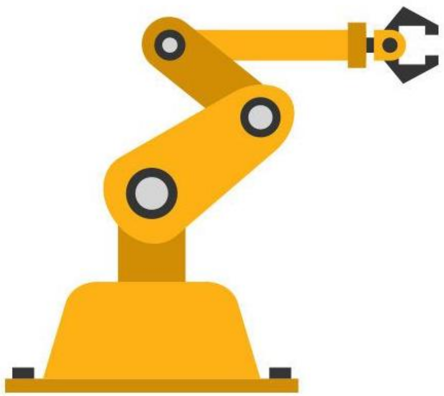
Author Alberto Archetti

Advisor Prof. Matteo Matteucci

Co-advisor Eng. Marco Cannici

Planning

Find the best sequence of actions to reach a goal.



Robotic motion

Games

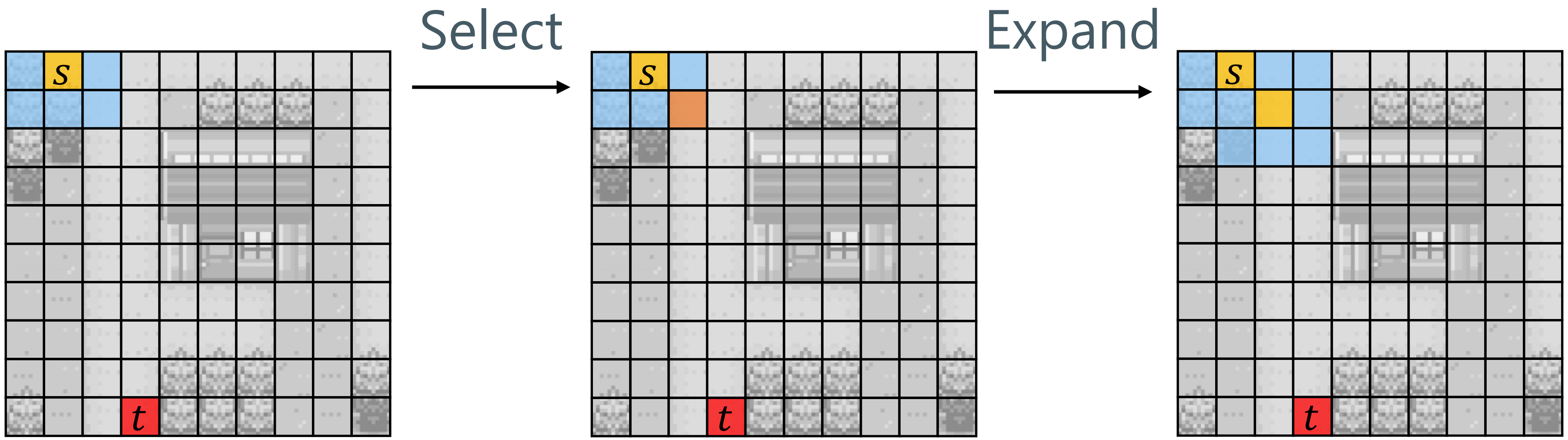
Navigation

[Russel et al., 1995]

The A* algorithm

Optimally efficient heuristic-based search algorithm.

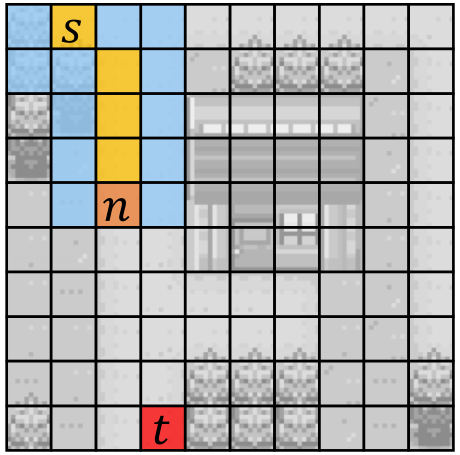
It searches for a path from the source node to the target node.



Priority measure

A*'s priority measure for node expansion:

$$F(n) = G(n) + H(n).$$



$G(n)$: exact cost between s and n.

$H(n)$: estimated cost between n and t.

[Russel et al., 1995]

Admissibility

$H(n)$ is admissible when it never overestimates the cost between n and t.

If $H(n)$ is admissible, A* is optimal and no other algorithm is more efficient.

[Russel et al., 1995]

A* planning cons

The optimal path may take exponential time to be found.

Heuristic design is non-trivial and domain-dependent.

[Hansen et al., 2007; Archetti et al., 2021]

A* planning cons

The optimal path may take exponential time to be found.

↓

Tradeoff planning accuracy for planning efficiency.

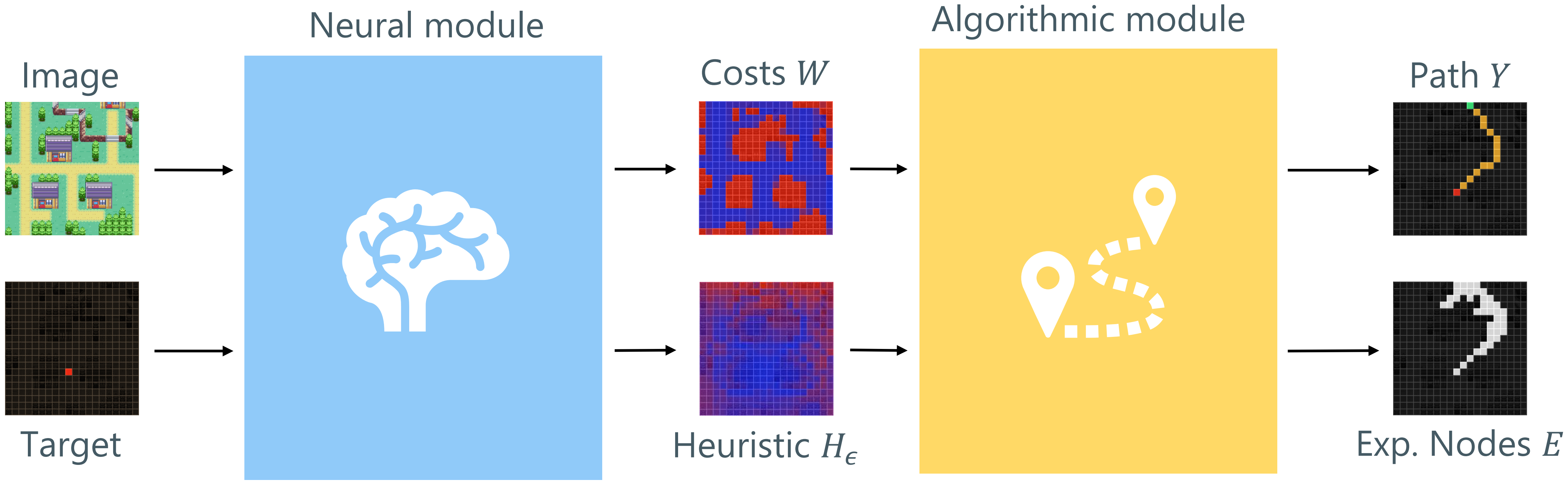
Heuristic design is non-trivial and domain-dependent.

↓

Predict graph labels from maps using deep learning.

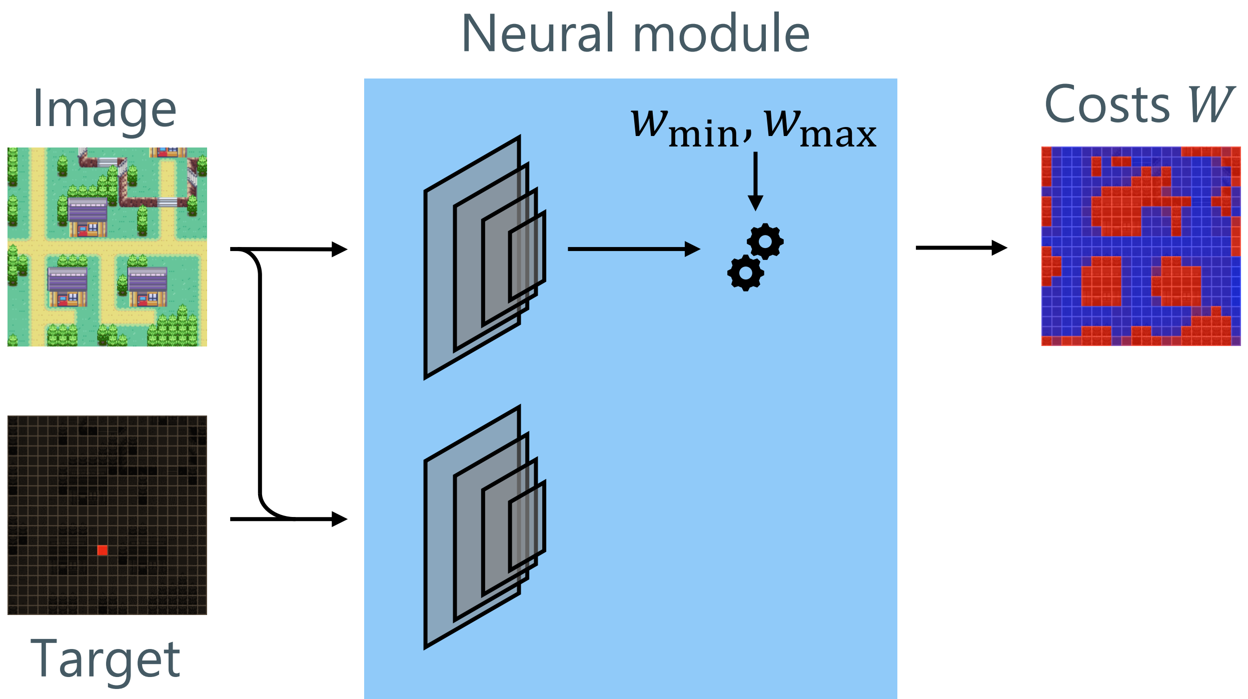
[Hansen et al., 2007; Archetti et al., 2021]

Neural Weighted A*'s idea



[Archetti et al., 2021]

Neural module



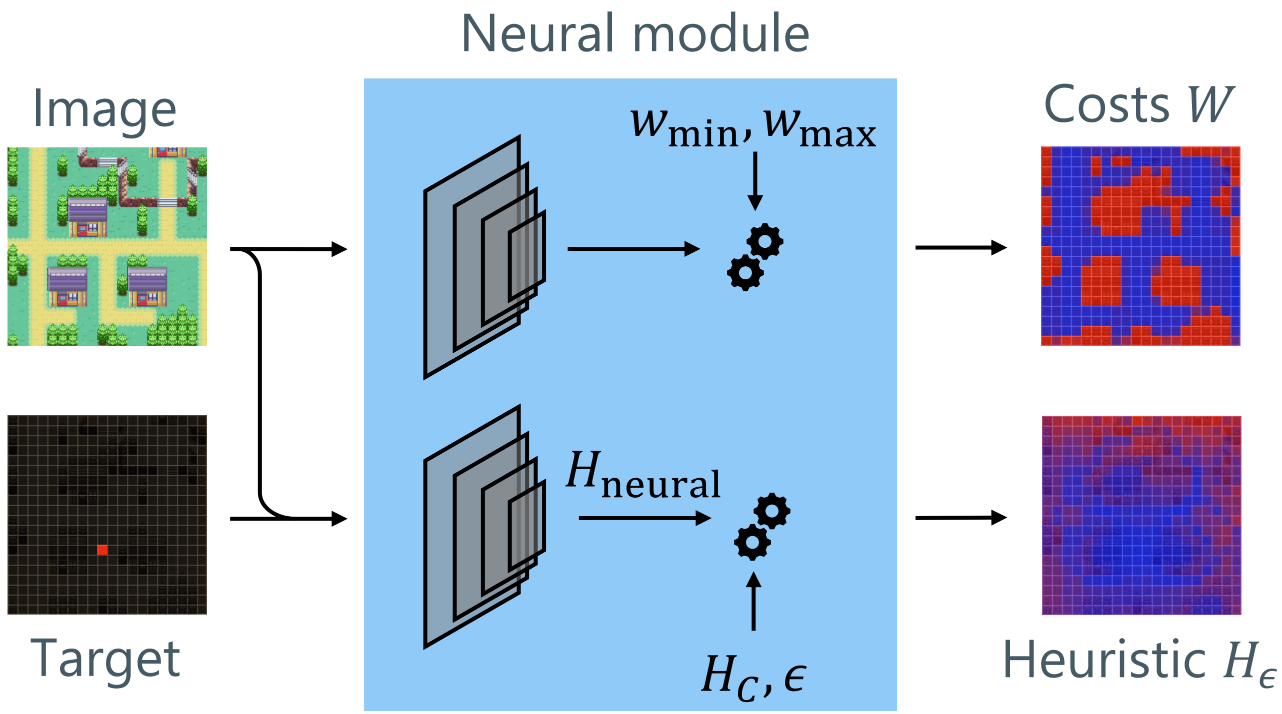
Accuracy-efficiency tradeoff key: relative scale between costs and heuristic.

The upper fully convolutional neural network predicts

$$W \in [w_{\min}, w_{\max}]$$

[Archetti et al., 2021]

Neural module



The lower fully convolutional neural network predicts

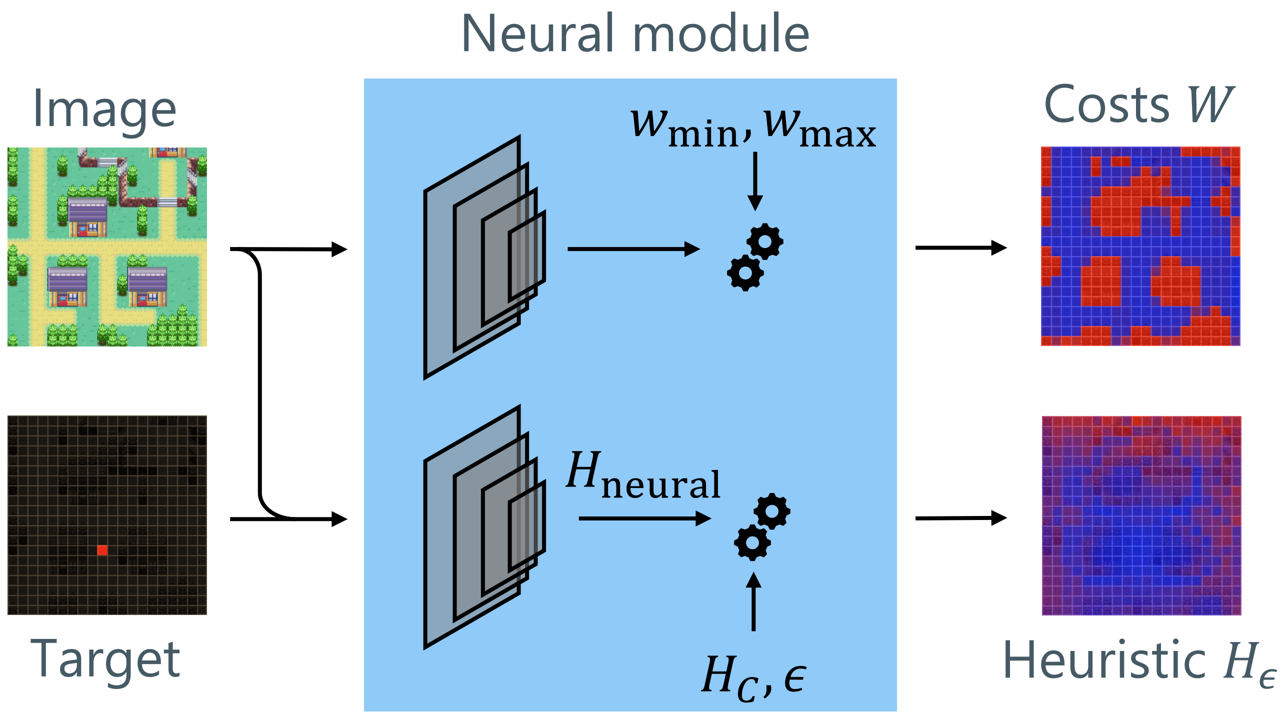
$$H_{\text{neural}} \in [0, 1]$$

Evaluate the admissible Chebyshev heuristic as

$$H_C(n) = w_{\min} \cdot D_C(n, t)$$

[Archetti et al., 2021]

Neural module



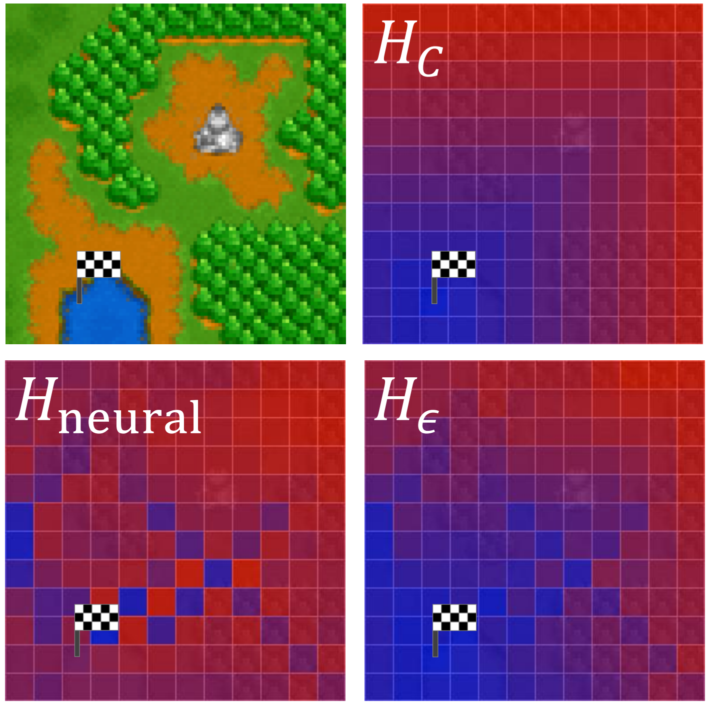
Define the tradeoff parameter ϵ.

Evaluate the final heuristic function as

$$H_\epsilon = (1 + \epsilon \cdot H_{\text{neural}}) \cdot H_C$$

[Archetti et al., 2021]

Heuristic scaling rationale



$$H_\epsilon = (1 + \epsilon \cdot H_{\text{neural}}) \cdot H_C$$

For $\epsilon = 0$, $H_\epsilon = H_C$, hence A* is optimal.

For $\epsilon > 0$,

- if $H_{\text{neural}}(n) \approx 0$, then $H_\epsilon(n) \approx H_C(n)$.
- if $H_{\text{neural}}(n) \approx 1$, then $H_\epsilon(n) \approx (1 + \epsilon) \cdot H_C(n)$.

[Archetti et al., 2021]

The Weighted A* method

Our formula comes from an A* extension, called Weighted A*.

It states that given the heuristic function $\alpha \cdot H_C$, then

$$\text{path cost } \leq \alpha \cdot \text{ optimal path cost.}$$

Since $H_\epsilon \leq (1+\epsilon) \cdot H_C$, then, for our architecture,

$$\text{path cost } \leq (1+\epsilon) \cdot \text{ optimal path cost.}$$

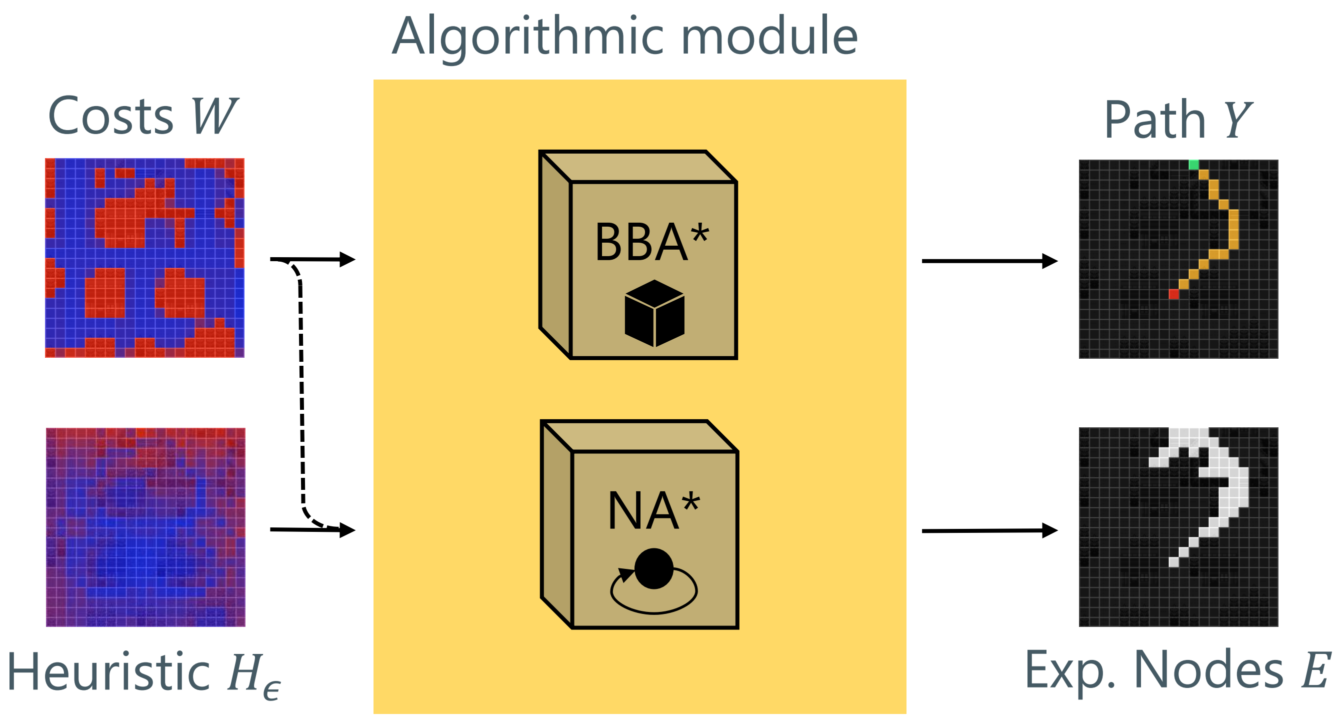
[Hansen et al., 2007; Archetti et al., 2021]

Structured learning

End-to-end differentiability achieved through two differentiable solvers:

- Black-Box A*
- Neural A*

We aim to encode different information in W and H_ϵ, while guaranteeing supervision on Y and E.



[Vlastelica et al., 2020; Yonetani et al., 2021; Archetti et al., 2021]

Training

Supervised learning on ground-truth path $\bar{Y}$:

$$\mathcal{L} = \alpha \cdot \mathcal{L}_H(\bar{Y}, Y) + \beta \cdot \mathcal{L}_H(\bar{Y}, E)$$

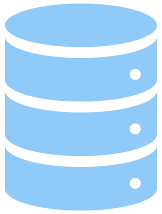
with $\mathcal{L}_H$ being the Hamming loss.

The idea is to force Y and $\bar{Y}$ to be as close as possible while minimizing the nodes in E.

[Vlastelica et al., 2020; Archetti et al., 2021]

Experiments

To validate our claims, we need three ingredients:



Datasets

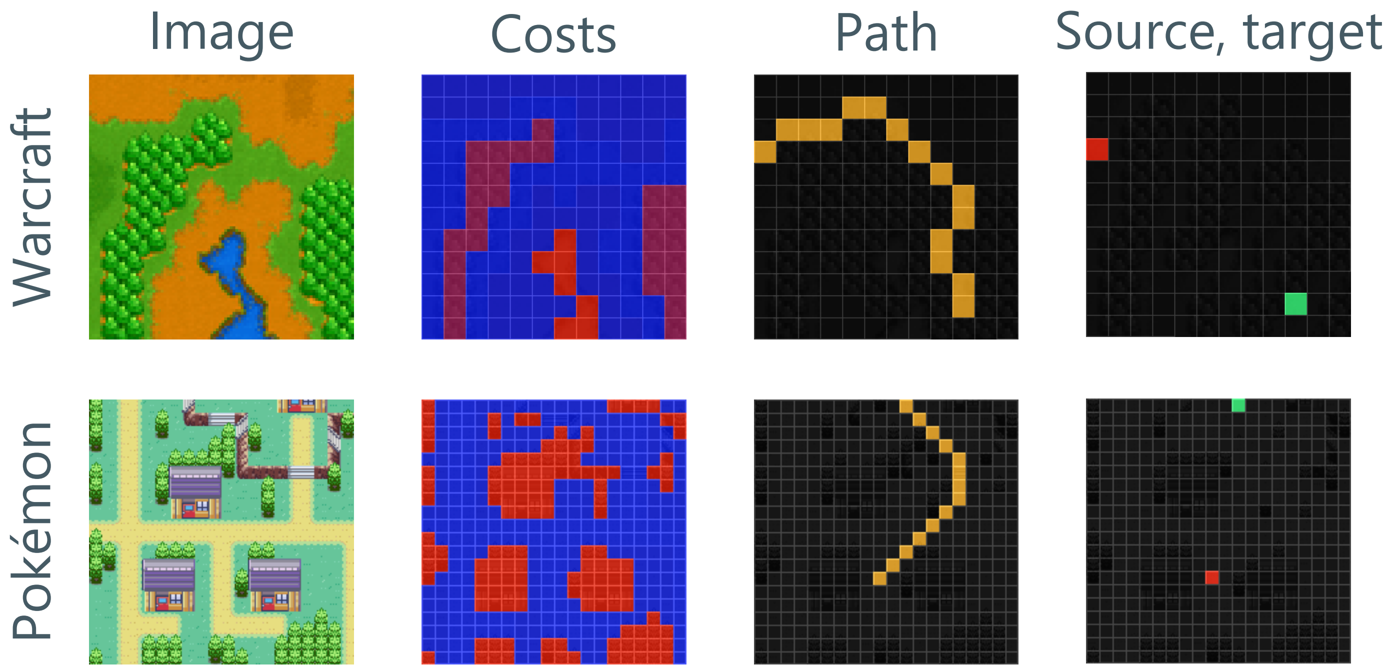
Metrics

Baselines

[Archetti et al., 2021]

Datasets

Tile-based planar navigation datasets.



[Vlastelica et al., 2020; Archetti et al., 2021]

Metrics

Accuracy:

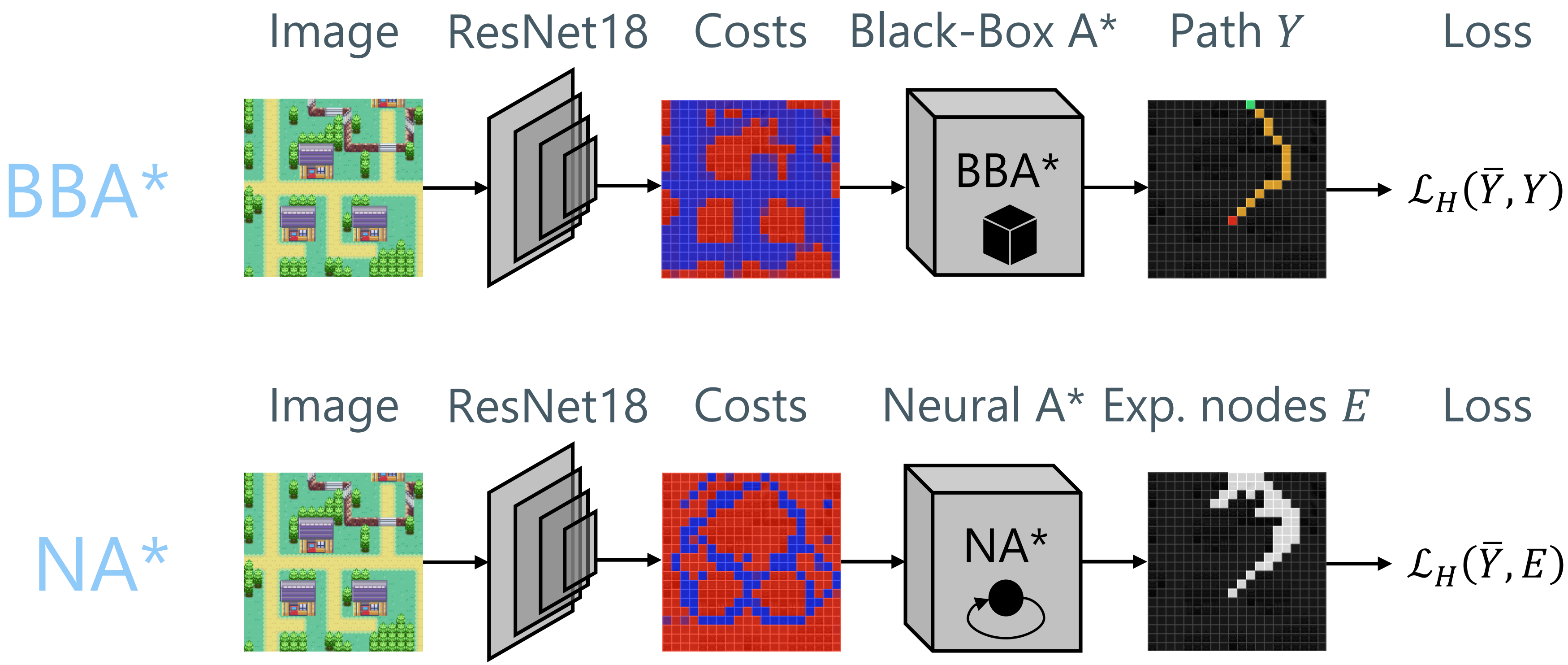
cost ratio = predicted path cost / true path cost

Efficiency:

expanded nodes = # of nodes expanded during the search

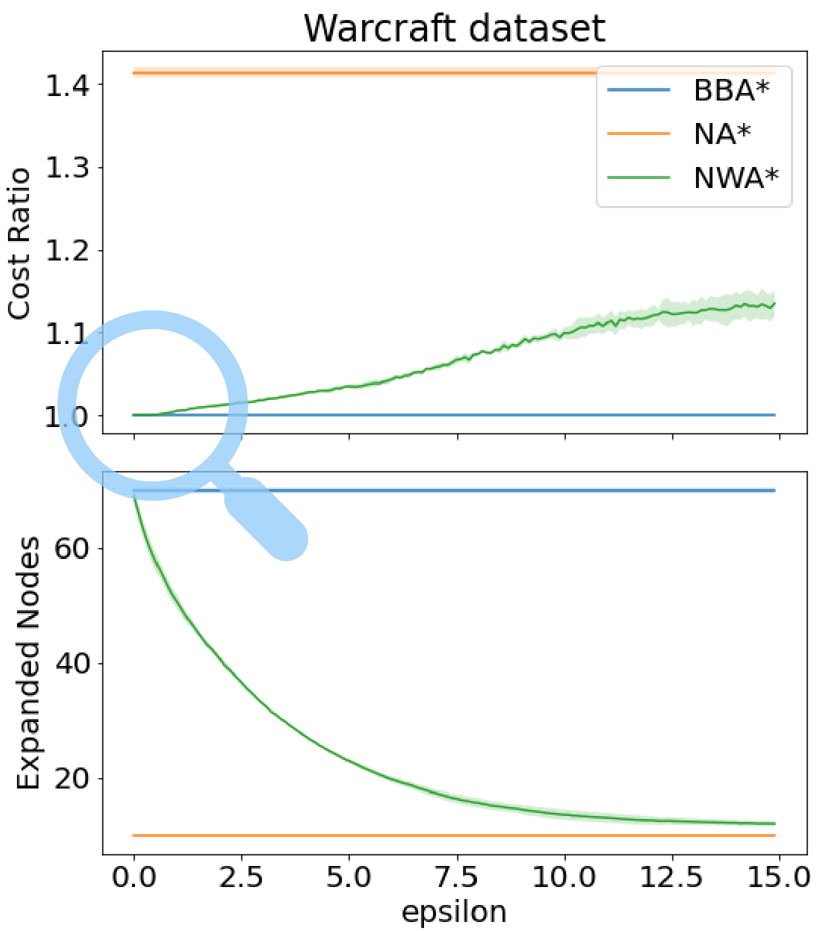
[Vlastelica et al., 2020; Archetti et al., 2021]

Baselines



[Vlastelica et al., 2020; Yonetani et al., 2021; Archetti et al., 2021]

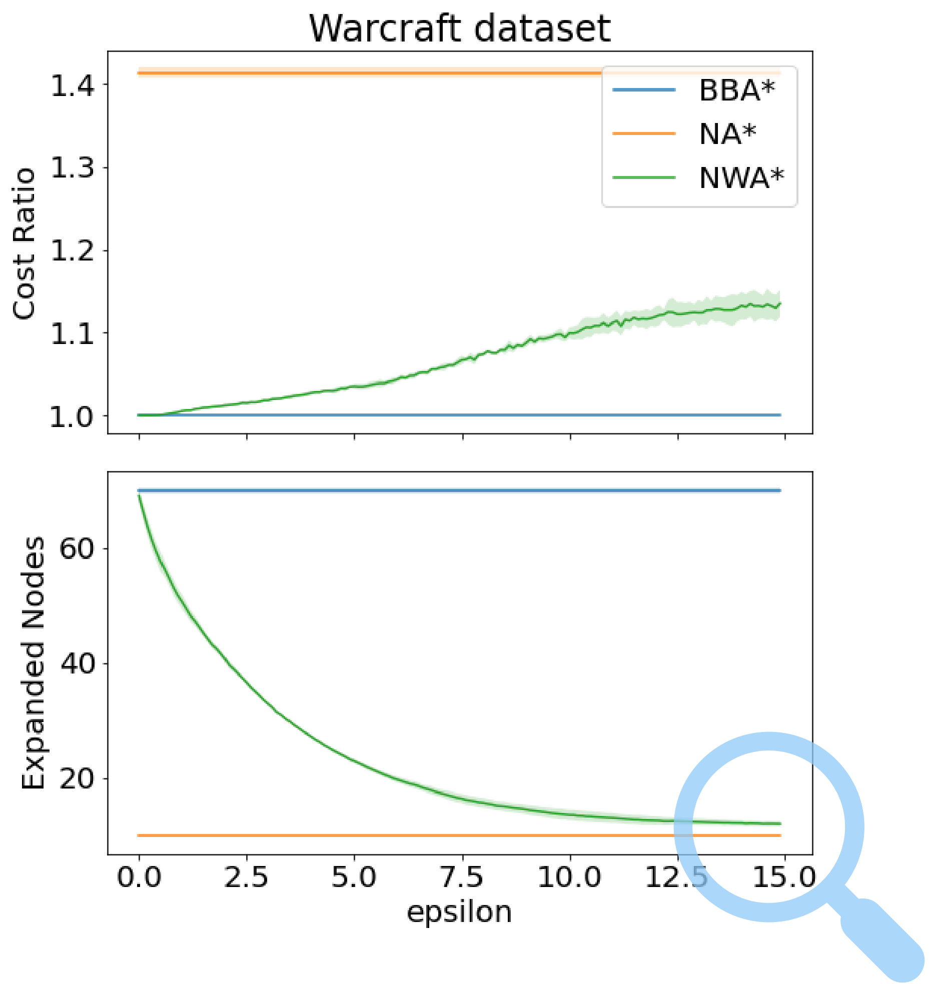
Results, Warcraft data



Low ϵ: NWA* as accurate as BBA* (cost ratio ≈ 1).

[Vlastelica et al., 2020; Yonetani et al., 2021; Archetti et al., 2021]

Results, Warcraft data

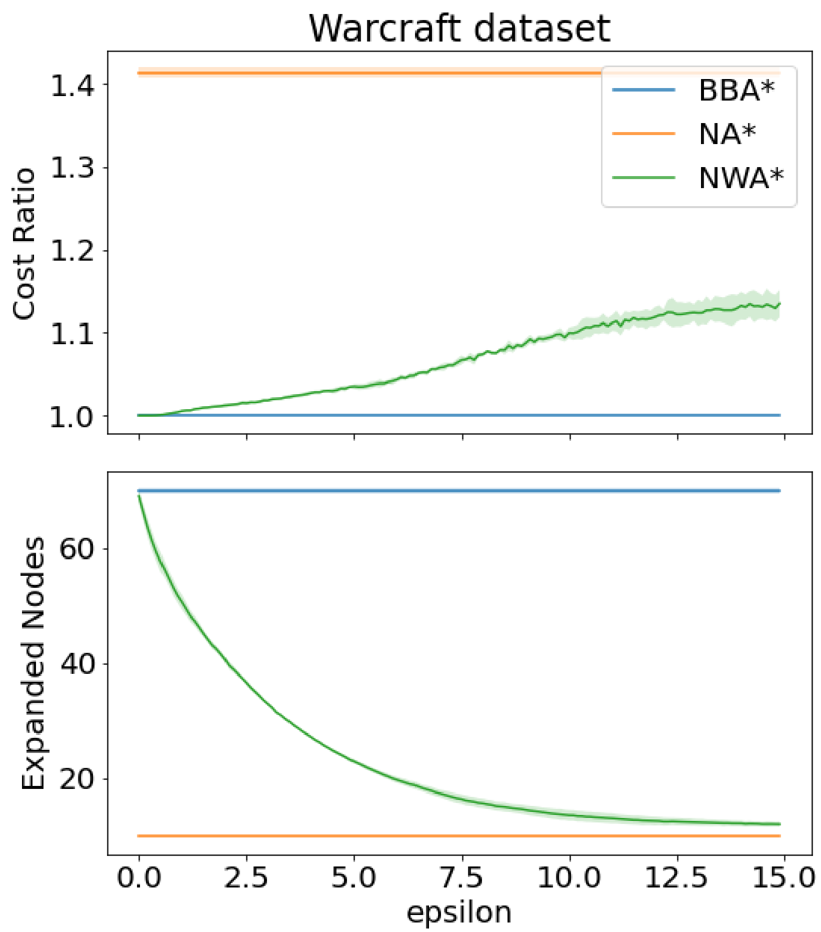


Low ϵ: NWA* as accurate as BBA* (cost ratio ≈ 1).

High ϵ: NWA* as efficient as NA* (expanded nodes ≈ 10) with better cost ratio.

[Vlastelica et al., 2020; Yonetani et al., 2021; Archetti et al., 2021]

Results, Warcraft data



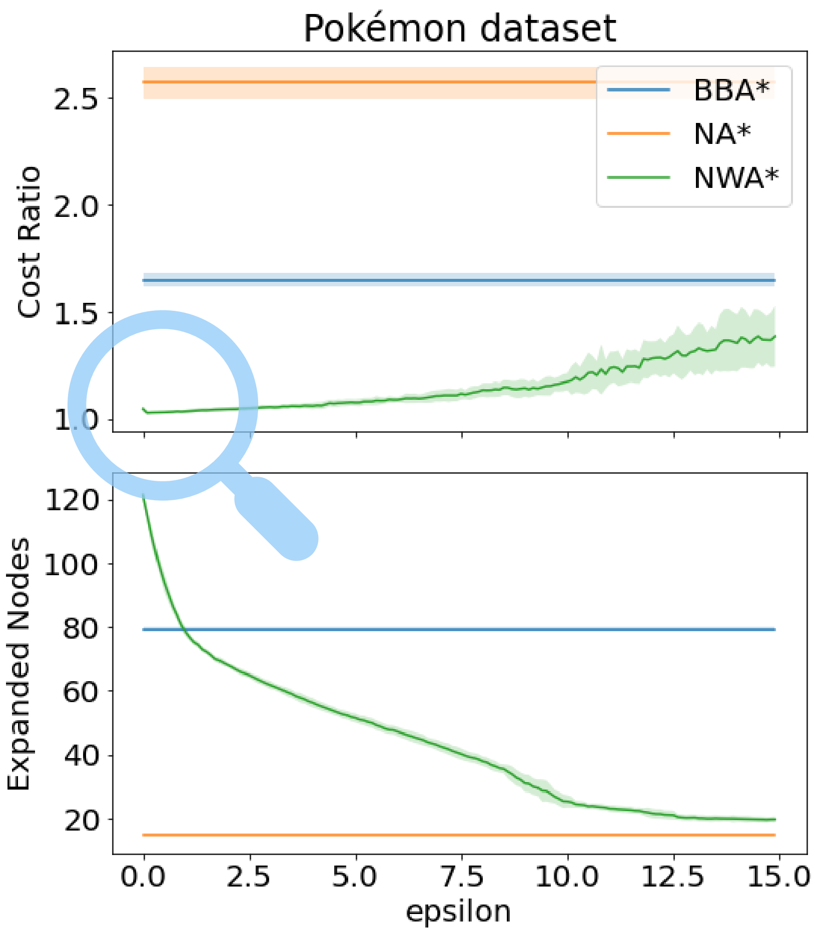
Low ϵ: NWA* as accurate as BBA* (cost ratio ≈ 1).

High ϵ: NWA* as efficient as NA* (expanded nodes ≈ 10) with better cost ratio.

NWA* can imitate the behavior of the baselines providing a principled, smooth tradeoff between accuracy and efficiency.

[Vlastelica et al., 2020; Yonetani et al., 2021; Archetti et al., 2021]

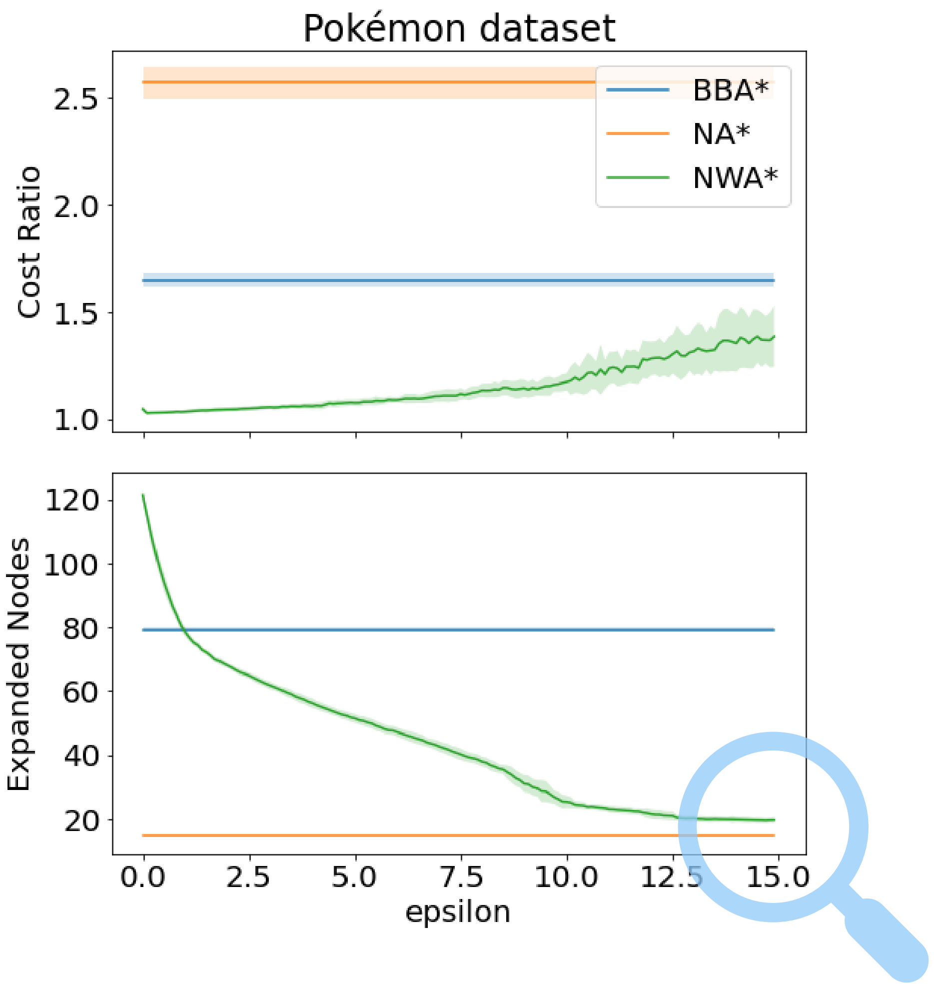
Results, Pokémon data



Low ϵ: NWA* is the most accurate (cost ratio ≈ 1).

[Vlastelica et al., 2020; Yonetani et al., 2021; Archetti et al., 2021]

Results, Pokémon data

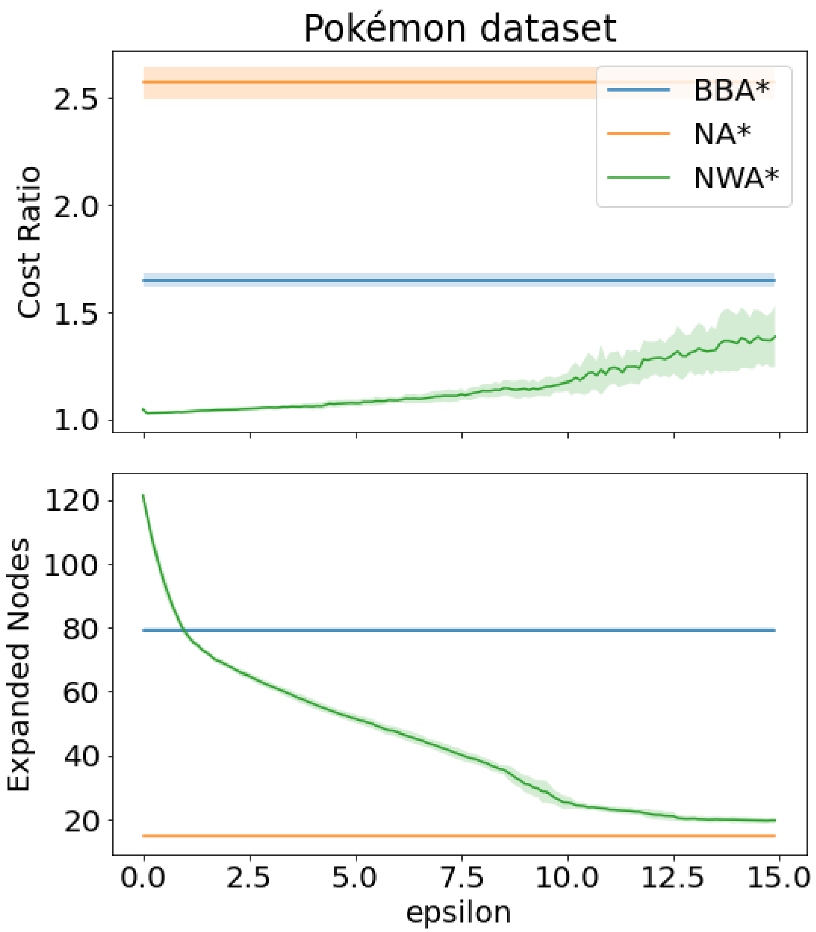


Low ϵ: NWA* is the most accurate (cost ratio ≈ 1).

High ϵ: NWA* as efficient as NA* (expanded nodes ≈ 20), with better cost ratio.

[Vlastelica et al., 2020; Yonetani et al., 2021; Archetti et al., 2021]

Results, Pokémon data



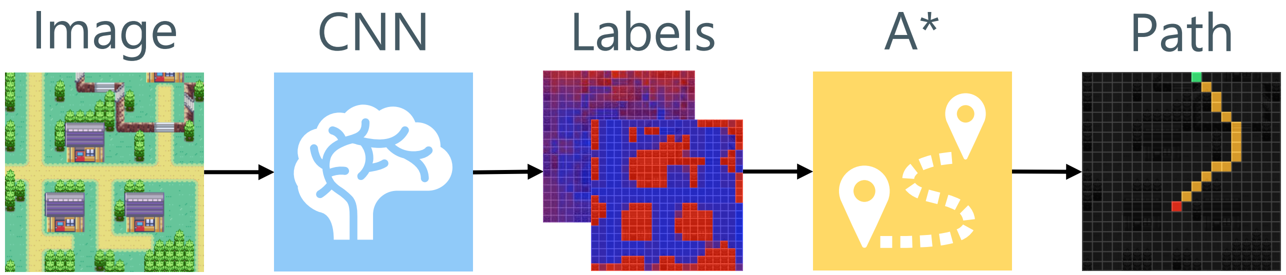
Low ϵ: NWA* is the most accurate (cost ratio ≈ 1).

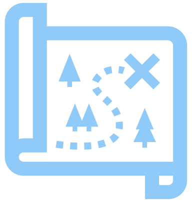
High ϵ: NWA* as efficient as NA* (expanded nodes ≈ 20), with better cost ratio.

NWA* can outperform the baselines in a complex scenario.

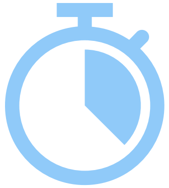
[Vlastelica et al., 2020; Yonetani et al., 2021; Archetti et al., 2021]

Conclusions





Enable planning on raw, unlabeled images.



Tradeoff planning accuracy for efficiency.

Propose a novel, tile-based dataset.

References

- Russell, S.J., Norvig, P.: Artificial intelligence - a modern approach: the intelligent agent book (1995)
- Hansen, E.A., Zhou, R.: Anytime heuristic search (2007)
- Vlastelica, M., Paulus, A., Musil, V., Martius, G., Rolnek, M.: Differentiation of blackbox combinatorial solvers (2020)
- Yonetani, R., Taniai, T., Barekatain, M., Nishimura, M., Kanezaki, A.: Path planning using neural A* search (2021)
- Archetti, A., Cannici, M., Matteucci, M.: Neural Weighted A*: Learning Graph Costs and Heuristics with Differentiable Anytime A* (2021, submitted)