Outline

1 Background
   - Planning
   - Machine Learning
   - Artificial Neural Networks

2 Learning a Heuristic Function
   - Search Strategy
   - Walk Strategy
   - Prediction Strategy

3 Results
   - Search Strategy
   - Walk Strategy
   - Prediction Strategy
   - Discussion

4 Conclusion and Future Work
Background
Planning task:

- Variables
- States assign values to variables
  - Initial state $s_0$
  - Goal states $s_*$
- Operators have preconditions and effects
Heuristic Search

- Heuristic search in state space
- Heuristic function
  - Estimates cost to nearest goal
- Use heuristic function to guide the search towards the goal
Machine Learning

Machine Learning . . .

...is the task of learning from data and make predictions on data.

- Set of input values $\vec{x}$
- Unknown target function $f^*: \vec{x} \rightarrow \vec{y}$
- Set of function hypotheses $F = \{ f \mid f: \vec{x} \rightarrow \vec{y}\}$

Goal

Find a function $f$ that approximates $f^*$ the best.
Artificial Neural Network

- Inspired by the brain
- Can predict data
- Needs to be trained
From Regression to Heuristic Function

- Regression
  - Find a function $f : \vec{x} \rightarrow \vec{y}$
- Heuristic function
  - Function $h : \vec{v} \rightarrow h_{val}$

$f = h$

- $\vec{x} = \vec{v}$
- $\vec{y} = h_{val}$

- $\vec{x}$ use variable values of any state $s$
- $\vec{y}$ use distance from state $s$ to the goal
Learning a Heuristic Function
Learning a Heuristic Function

- Use ANN as heuristic function
- Train ANN with *back-propagation*
- Generate training set for *back-propagation*
  - Search Strategy
  - Walk Strategy
  - Prediction Strategy
Generate the Training Set with Search Strategy

- Chose random goal state $s_*$
- Perform random walk starting at $s_*$
- Search from random walk endpoint
- Add every state from the solution path to the training set
Generate the Training Set with Walk Strategy

- Choose random goal state $s_*$
- Perform random walk starting at $s_*$
- Add every state on the random walk to the training set
Generate the Training Set with Prediction Strategy

- Chose random goal state $s_*$
- Perform random walk starting at $s_*$
- Add every state on the random walk to the training set
- Use a solution cost predictor to estimate distance
Results
Baseline

Learning Heuristic Functions in Classical Planning (LHFCP)

<table>
<thead>
<tr>
<th></th>
<th>blind-search</th>
<th>LHFCP</th>
<th>$h^{FF}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage</td>
<td>680</td>
<td>693</td>
<td>1308</td>
</tr>
<tr>
<td>Expansions</td>
<td>53369.73</td>
<td>24990.02</td>
<td>282.75</td>
</tr>
<tr>
<td>Search time</td>
<td>1.31</td>
<td>1.62</td>
<td>0.25</td>
</tr>
<tr>
<td>Total time</td>
<td>1.35</td>
<td>8.66</td>
<td>0.27</td>
</tr>
</tbody>
</table>

- Slightly better than \textit{blind-search}
- High total time
- Higher search time than \textit{blind-search}
- Worse than $h^{FF}$
Performance against *blind-search*

### Expansions:

![Expansions Chart]

### Search time:

![Search time Chart]
Performance against $h^{FF}$

Expansions:

Search time:

![Graph showing performance against $h^{FF}$](imageURL)
## Expansions Gripper

<table>
<thead>
<tr>
<th>prob</th>
<th>blind-search</th>
<th>LHFCP</th>
<th>$h^{FF}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>prob01.pddl</td>
<td>253</td>
<td>110</td>
<td>123</td>
</tr>
<tr>
<td>prob02.pddl</td>
<td>1853</td>
<td>993</td>
<td>1413</td>
</tr>
<tr>
<td>prob03.pddl</td>
<td>11773</td>
<td>10648</td>
<td>10735</td>
</tr>
<tr>
<td>prob04.pddl</td>
<td>68605</td>
<td>65716</td>
<td>66585</td>
</tr>
<tr>
<td>prob05.pddl</td>
<td>376829</td>
<td>370598</td>
<td>373347</td>
</tr>
<tr>
<td>prob06.pddl</td>
<td>1982461</td>
<td>1913835</td>
<td>1976941</td>
</tr>
<tr>
<td>prob07.pddl</td>
<td>10092541</td>
<td>10051648</td>
<td>10084311</td>
</tr>
</tbody>
</table>

- **LHFCP** performs better on domain **gripper** than $h^{FF}$
- Also on the domain **psr-small**
### Random Walk Length

<table>
<thead>
<tr>
<th></th>
<th>rwl-20</th>
<th>default</th>
<th>rwl-100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage</td>
<td>687</td>
<td>693</td>
<td>690</td>
</tr>
<tr>
<td>Expansions</td>
<td>29221.89</td>
<td>29205.89</td>
<td>27534.29</td>
</tr>
<tr>
<td>Search time</td>
<td>1.87</td>
<td>1.84</td>
<td>1.79</td>
</tr>
<tr>
<td>Total time</td>
<td>9.32</td>
<td>9.49</td>
<td>10.02</td>
</tr>
<tr>
<td>Training set size</td>
<td>396.76</td>
<td>457.66</td>
<td>519.24</td>
</tr>
</tbody>
</table>

- Longer random walks produce bigger **training set**
- Longer random walks produce slightly better heuristic
- Longer random walks need more **setup time**
Number of Random Walks

<table>
<thead>
<tr>
<th></th>
<th>nrw-100</th>
<th>default</th>
<th>nrw-500</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage</td>
<td>701</td>
<td>693</td>
<td>675</td>
</tr>
<tr>
<td>Expansions</td>
<td>26726.12</td>
<td>25701.05</td>
<td>24337.35</td>
</tr>
<tr>
<td>Search time</td>
<td>1.76</td>
<td>1.73</td>
<td>1.70</td>
</tr>
<tr>
<td>Total time</td>
<td>7.79</td>
<td>8.95</td>
<td>11.79</td>
</tr>
<tr>
<td>Training set size</td>
<td>311.93</td>
<td>466.80</td>
<td>912.01</td>
</tr>
</tbody>
</table>

- More random walks produce bigger **training set**
- More random walks produce slightly better heuristic
- More random walks need more **setup time**
Training Set Size for Number of Random Walks

- Not possible to generate **training set** on each domain
- Size of **training set** does not scale linear on every domain
### ANN-Topology

<table>
<thead>
<tr>
<th></th>
<th>topo-n-20-1</th>
<th>default</th>
<th>topo-n-100-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage</td>
<td>679</td>
<td>693</td>
<td>689</td>
</tr>
<tr>
<td>Expansions</td>
<td>35034.57</td>
<td>28690.82</td>
<td>26111.88</td>
</tr>
<tr>
<td>Search time</td>
<td>1.60</td>
<td>1.81</td>
<td>2.31</td>
</tr>
<tr>
<td>Total time</td>
<td>7.86</td>
<td>9.43</td>
<td>12.05</td>
</tr>
<tr>
<td>Training set size</td>
<td>454.60</td>
<td>453.10</td>
<td>452.80</td>
</tr>
</tbody>
</table>

- More neurons produce slightly better heuristic
- More neurons need more **setup time**
- More neurons need more **search time**
Different Initial Heuristic Functions $h_0$

<table>
<thead>
<tr>
<th></th>
<th>$h_0$-blind-search</th>
<th>$h_0$-FF</th>
<th>$h_0$-ipdb</th>
<th>$h_0$-lm-cut</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage</td>
<td>683</td>
<td>705</td>
<td>604</td>
<td>699</td>
</tr>
<tr>
<td>Expansions</td>
<td>20814.37</td>
<td>19795.51</td>
<td>21012.24</td>
<td>20664.96</td>
</tr>
<tr>
<td>Search time</td>
<td>1.48</td>
<td>1.39</td>
<td>1.41</td>
<td>1.40</td>
</tr>
<tr>
<td>Total time</td>
<td>10.79</td>
<td>5.53</td>
<td>51.03</td>
<td>6.46</td>
</tr>
<tr>
<td>Training set size</td>
<td>181.35</td>
<td>295.55</td>
<td>242.87</td>
<td>234.74</td>
</tr>
</tbody>
</table>

- **FF** produces the strongest heuristic
- **blind-search** has the smallest **training set**
- **ipdb** has highest **setup time**
Walk Strategy

<table>
<thead>
<tr>
<th></th>
<th>search-strategy</th>
<th>walk-strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage</td>
<td>693</td>
<td>692</td>
</tr>
<tr>
<td>Expansions</td>
<td>25367.74</td>
<td>25153.76</td>
</tr>
<tr>
<td>Search time</td>
<td>1.65</td>
<td>1.61</td>
</tr>
<tr>
<td>Total time</td>
<td>8.80</td>
<td>7.82</td>
</tr>
<tr>
<td>Training set size</td>
<td>453.09</td>
<td>1850.41</td>
</tr>
</tbody>
</table>

- Both approaches perform about the same
- Lower total time with walk-strategy
- Bigger training set with walk-strategy
Prediction Strategy

<table>
<thead>
<tr>
<th>search-strategy</th>
<th>prediction-strategy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coverage</td>
<td>49</td>
</tr>
<tr>
<td>Expansions</td>
<td>21.21</td>
</tr>
<tr>
<td>Search time</td>
<td>0.10</td>
</tr>
<tr>
<td>Total time</td>
<td>0.36</td>
</tr>
<tr>
<td>Training set size</td>
<td>274.89</td>
</tr>
</tbody>
</table>

- Only executed on domain **psr-small**
- Lower **coverage** with **prediction-strategy**
- Higher **total time** with **prediction-strategy**
- Bigger **training set** with **prediction-strategy**
Discussion

- Random walk length affects training set size
- Number of random walks affects training set size
- ANN-topology affect total time
- Initial heuristic should have small setup time
- Search can be omitted
- Solution cost predictor too resource hungry
Conclusion and Future Work
Conclusion

- Has high setup time
- Depends on many parameters
- Inverse operators
- Domain-independence not possible
- Can be better than $h^{FF}$ on some domains

Future Work

- Adopt parameters to current problem
- Use other features
- Use other machine learning techniques
Question?