CIS730
PYSC2 Game Playing Agent

John Boyington

12/13/18

Project Description

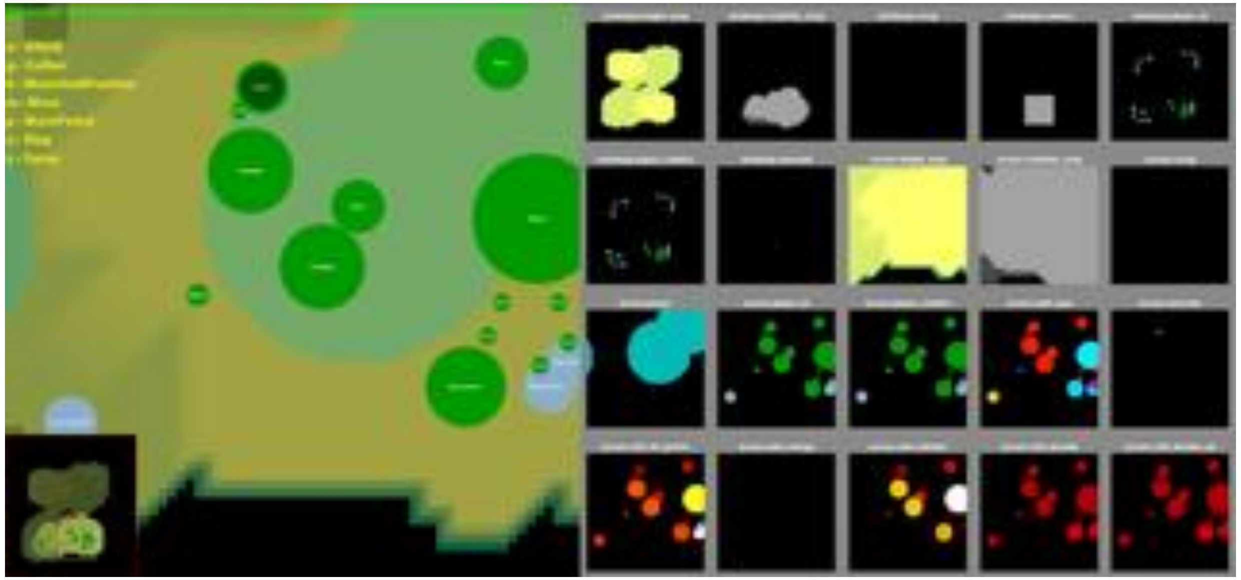
- Train an Artificial Neural Network (ANN) to mimic the behavior of a scripted bot in the pysc2 minigame ‘DefeatRoaches’.
- https://github.com/johnboyington/cis730_project

About the Game

- Real Time Strategy
- Incredibly Complex
- Open problem in Artificial Intelligence

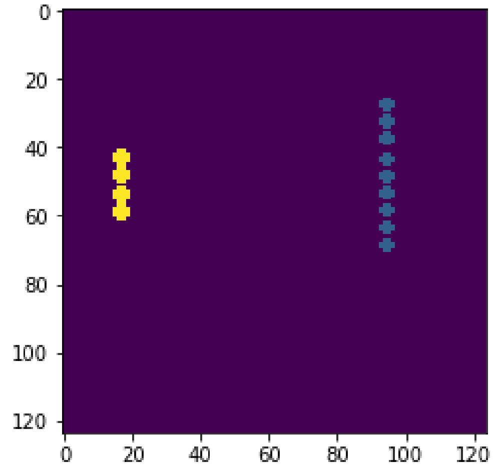
PYSC2 API – Feature Screens, Units, etc.

- Feature Screens – Provides spatial information about game (visible area, buildings, unit health, unit type, etc.).
- Feature Units – Provides info about onscreen units, (position, damage, health, etc.)



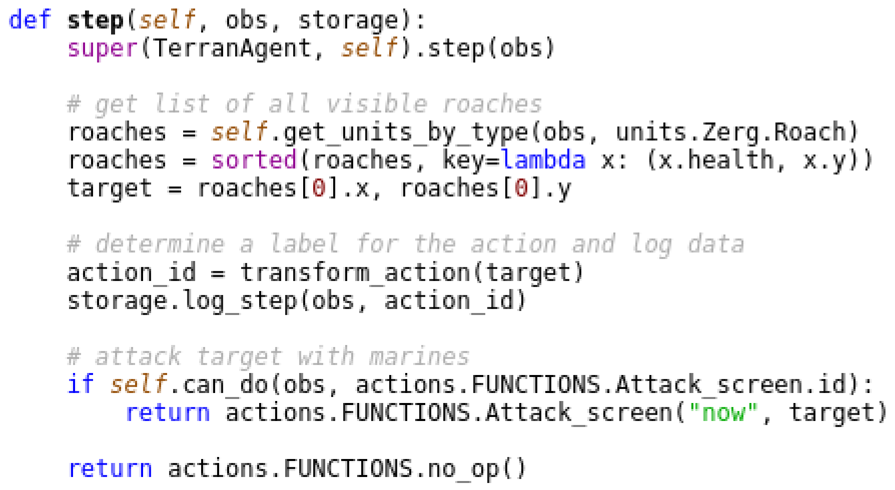
PYSC2 API - DefeatRoaches

- Initial State – 9 Marines and 4 Roaches placed in random locations in a vertical formation on opposite sides of screen.
- Rewards – (+10) for Roach Defeated, (-1) for Marine Defeated
- End Cond. – 120 seconds, or all Marines defeated.
- Additional Notes – No limit to visibility, no camera movement.



Scripted Bot

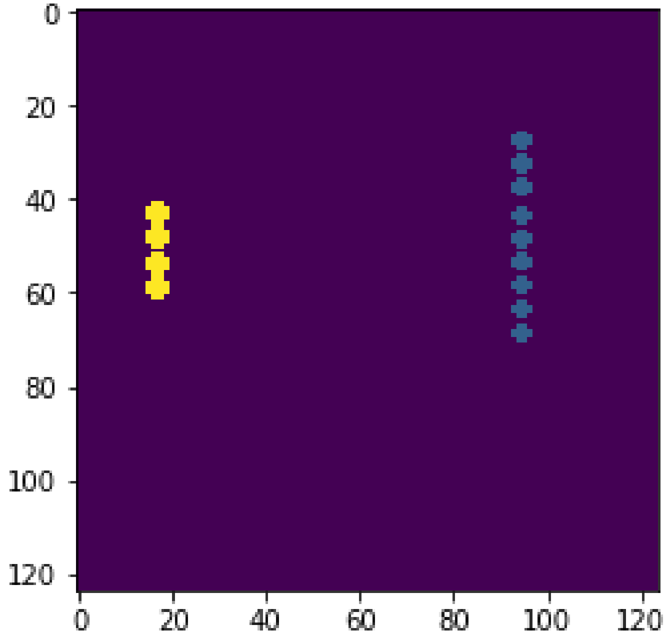
- Limited to only attacking
- Target selected based on:
  - Lowest Health
  - Break ties with y-component of position

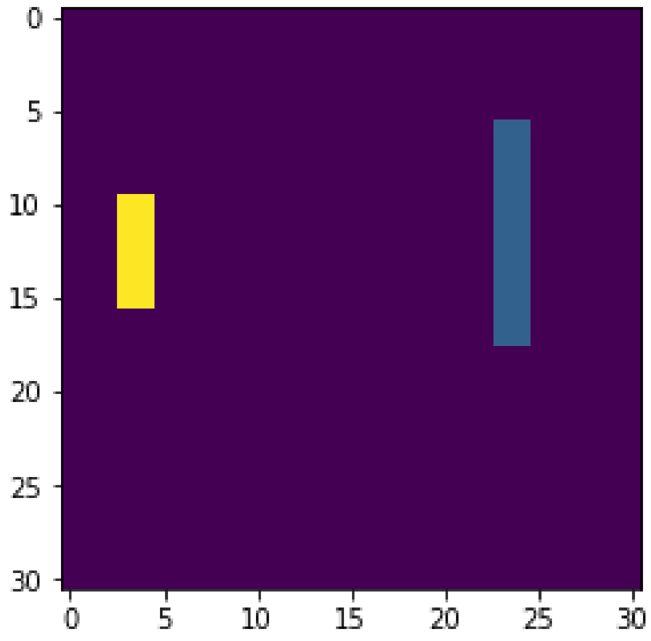


```python
def step(self, obs, storage):
    super(TerranAgent, self).step(obs)

    # get list of all visible roaches
    roaches = self.get_units_by_type(obs, units.Zerg.Roach)
    roaches = sorted(roaches, key=lambda x: (x.health, x.y))
    target = roaches[0].x, roaches[0].y

    # determine a label for the action and log data
    action_id = transform_action(target)
    storage.log_step(obs, action_id)

    # attack target with marines
    if self.can_do(obs, actions.FUNCTIONS.Attack_screen.id):
        return actions.FUNCTIONS.Attack_screen("now", target)

    return actions.FUNCTIONS.no_op()
```

Data Preprocessing

- Removal of friendly units
- Value scaling
- Image compression (124x124 to 31x31)



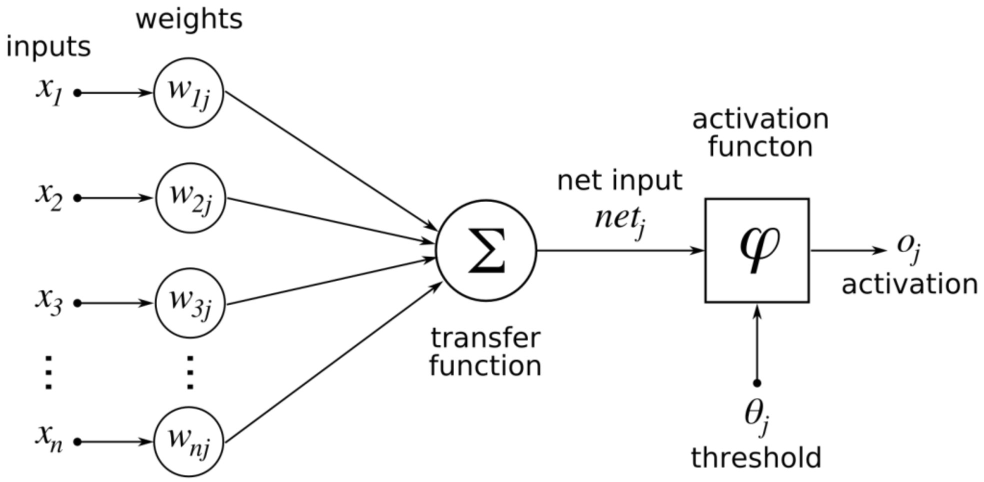


Action Space Formulation

- Continuous space (120x100 units)
- Discretized to 30x25 attack locations
- Flattened to 750 unique values and enumerated (for classification)

Training Data & ANN Parameters

- 49,247 training images (shuffled)
- ANN (tensorflow’s implementation of Keras API)
  - Input layer (length=1922)
  - Output layer (length=750, activation=‘relu’)
- Epochs = 3
- Batch Size = 64



Results & Performance

- Scripted Bot avg score 160.4
- ANN avg score -2.8
  - Average Error 1.2166e-3

Discussion & Future Work

- Limited dataset
- Problem in action space formulation

References

- Vinyals, Oriol, et al. "Starcraft ii: A new challenge for reinforcement learning." *arXiv preprint arXiv:1708.04782*(2017).
- Russell, Stuart J., and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.
- Chollet, François. "Keras: Deep learning library for theano and tensorflow.(2015)." *There is no corresponding record for this reference* (2015).
- Ontanón, Santiago, et al. "A survey of real-time strategy game AI research and competition in StarCraft." *IEEE Transactions on Computational Intelligence and AI in games* 5.4 (2013): 293-311.

References

- Liu, Siming, Sushil J. Louis, and Christopher A. Ballinger. "Evolving effective microbehaviors in real-time strategy games." *IEEE Transactions on Computational Intelligence and AI in Games* 8.4 (2016): 351-362.