Research plan

Fleischmann – Nassar joint meeting

Andrea Pierré

May 17, 2024

Brown University

Outline

1. Hypothesis & directions
2. Experiments & expected results
3. Roadmap

Outline

1. Hypothesis & directions
2. Experiments & expected results
3. Roadmap

What do we want to know?

- Understand what the network learns $\rightarrow$ What function does it learns?
- How the constrains of the task affect learning & the representations learned?
- Does the network learn something related to the real neurons? (million **$$$** question)

What do we want to know?

- Understand what the network learns $\rightarrow$ What function does it learns?
- How the constrains of the task affect learning & the representations learned?
- Does the network learn something related to the real neurons? (million **$$$** question)

What do we want to know?

- Understand what the network learns $\rightarrow$ What function does it learns?
- How the constrains of the task affect learning & the representations learned?
- Does the network learn something related to the real neurons? (million **$$$** question)

Compositional hypothesis

- Does the network learn a generalizable policy?
  $\rightarrow$ How to test it?
  - From indexed locations (current) to coordinate system
  - Merged actions space

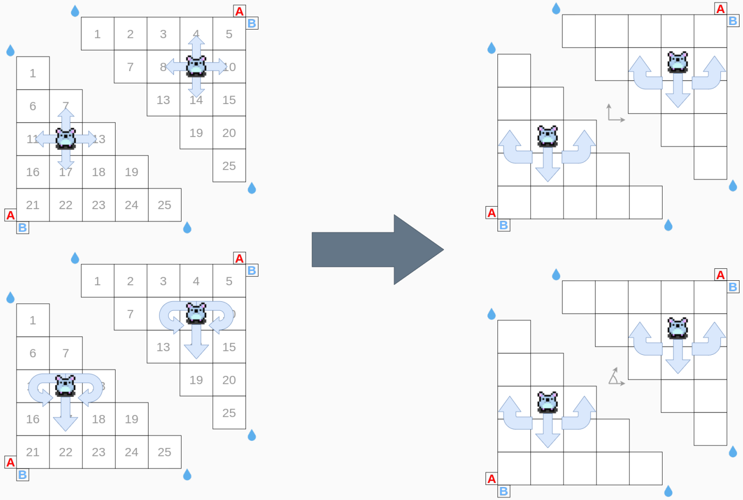
Compositional hypothesis

- Does the network learn a generalizable policy?
  $\rightarrow$ How to test it?
  - From indexed locations (current) to coordinate system
  - Merged actions space

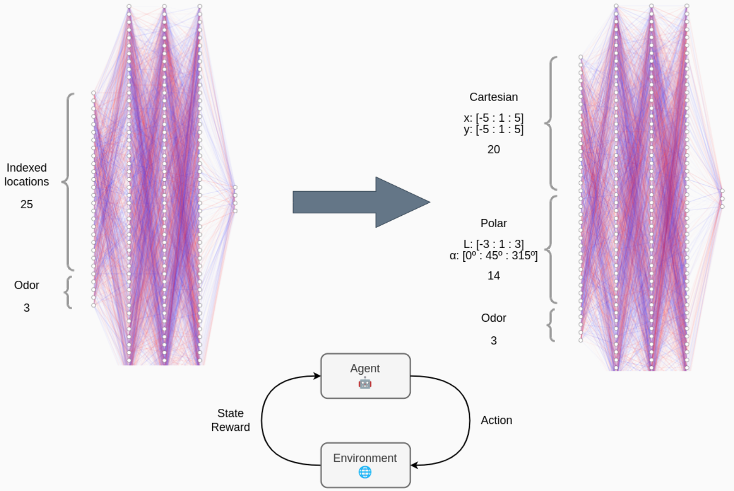
Compositional hypothesis

- Does the network learn a generalizable policy?
  $\rightarrow$ How to test it?
    - From indexed locations (current) to coordinate system
    - Merged actions space

Compositional hypothesis



Implementation



Outline

1. Hypothesis & directions
2. Experiments & expected results
3. Roadmap

1) How training impacts the representations learned?

- Feed both coordinates information (Cartesian & polar) to the input layer (+ merge actions spaces in a common one)
- Train on left/right task $\rightarrow$ we expect the weights are close to zero on Cartesian representation?
- Train on east/west task $\rightarrow$ we expect the weights are close to zero on polar representation?
- Not clear to me how to extract/define which neurons belong/contribute to Cartesian/polar representation

1) How training impacts the representations learned?

- Feed both coordinates information (Cartesian & polar) to the input layer (+ merge actions spaces in a common one)
- Train on left/right task $\rightarrow$ we expect the weights are close to zero on Cartesian representation?
- Train on east/west task $\rightarrow$ we expect the weights are close to zero on polar representation?
- Not clear to me how to extract/define which neurons belong/contribute to Cartesian/polar representation

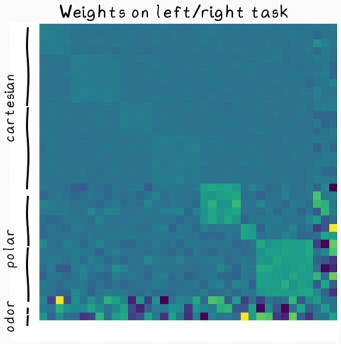
1) How training impacts the representations learned?

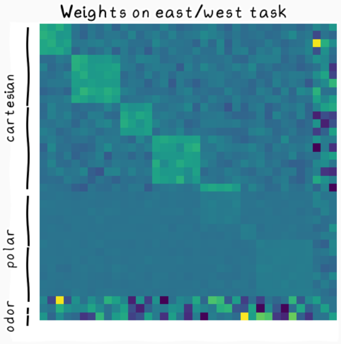
- Feed both coordinates information (Cartesian & polar) to the input layer (+ merge actions spaces in a common one)
- Train on left/right task $\rightarrow$ we expect the weights are close to zero on Cartesian representation?
- Train on east/west task $\rightarrow$ we expect the weights are close to zero on polar representation?
- Not clear to me how to extract/define which neurons belong/contribute to Cartesian/polar representation

1) How training impacts the representations learned?

- Feed both coordinates information (Cartesian & polar) to the input layer (+ merge actions spaces in a common one)
- Train on left/right task $\rightarrow$ we expect the weights are close to zero on Cartesian representation?
- Train on east/west task $\rightarrow$ we expect the weights are close to zero on polar representation?
- Not clear to me how to extract/define which neurons belong/contribute to Cartesian/polar representation

1) How training impacts the representations learned?





2) Does the network learn a coordinate system?

1. After training, move the population of agents in a translated coordinate system $\rightarrow$ we expect the population of agents to be able to solve the task with zero shot learning
2. Train with both coordinates information (Cartesian & polar), after training feed incorrect polar angles
   - On the left/right task $\rightarrow$ we expect the population of agents still solves the task consistently
   - On the east/west task $\rightarrow$ we expect the network won't converge to a stable policy (i.e all the agents don't solve the task consistently)

2) Does the network learn a coordinate system?

1. After training, move the population of agents in a translated coordinate system $\rightarrow$ we expect the population of agents to be able to solve the task with zero shot learning
2. Train with both coordinates information (Cartesian & polar), after training feed incorrect polar angles
   - On the left/right task $\rightarrow$ we expect the population of agents still solves the task consistently
   - On the east/west task $\rightarrow$ we expect the network won't converge to a stable policy (i.e all the agents don't solve the task consistently)

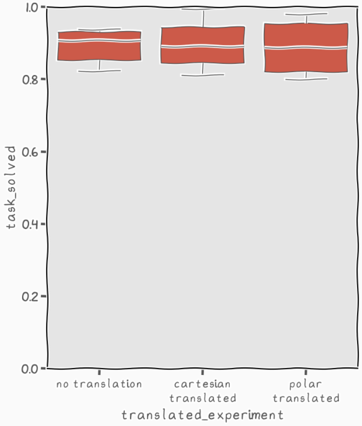
2) Does the network learn a coordinate system?

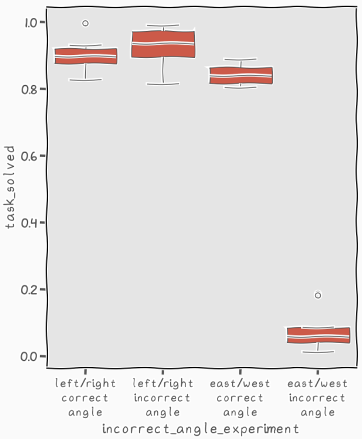
1. After training, move the population of agents in a translated coordinate system $\rightarrow$ we expect the population of agents to be able to solve the task with zero shot learning
2. Train with both coordinates information (Cartesian & polar), after training feed incorrect polar angles
   - On the left/right task $\rightarrow$ we expect the population of agents still solves the task consistently
   - On the east/west task $\rightarrow$ we expect the network won't converge to a stable policy (i.e all the agents don't solve the task consistently)

2) Does the network learn a coordinate system?

1. After training, move the population of agents in a translated coordinate system $\rightarrow$ we expect the population of agents to be able to solve the task with zero shot learning
2. Train with both coordinates information (Cartesian & polar), after training feed incorrect polar angles
    - On the left/right task $\rightarrow$ we expect the population of agents still solves the task consistently
    - On the east/west task $\rightarrow$ we expect the network won't converge to a stable policy (i.e all the agents don't solve the task consistently)

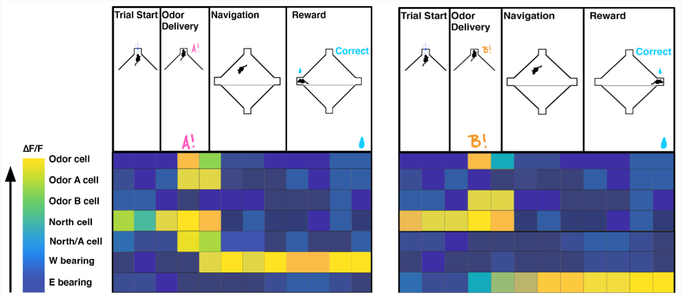
2) Does the network learn a coordinate system?





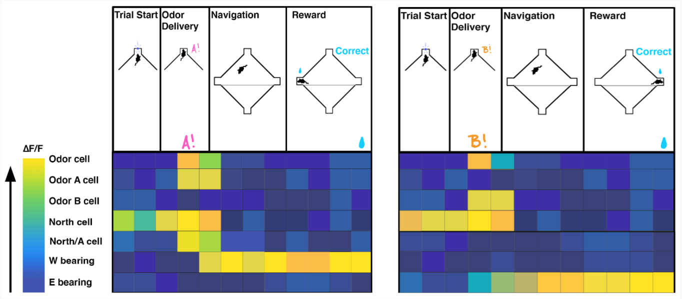
3) Conjunctive odor-place coding

- Train a population of agents, then after training, flip odor A and odor B in the task
- In general we expect to find a population of conjunctive neurons that get active with the combination of both odor and specific location
- Do the conjunctive cells get conserved or remapped? (Not clear to me, I'd expect they get remapped)



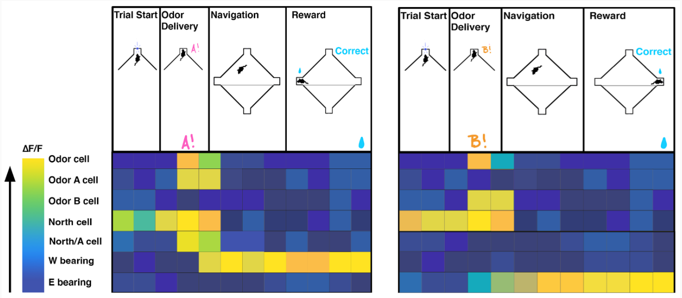
3) Conjunctive odor-place coding

- Train a population of agents, then after training, flip odor A and odor B in the task
- In general we expect to find a population of conjunctive neurons that get active with the combination of both odor and specific location
- Do the conjunctive cells get conserved or remapped? (Not clear to me, I'd expect they get remapped)



3) Conjunctive odor-place coding

- Train a population of agents, then after training, flip odor A and odor B in the task
- In general we expect to find a population of conjunctive neurons that get active with the combination of both odor and specific location
- Do the conjunctive cells get conserved or remapped? (Not clear to me, I'd expect they get remapped)



Outline

1. Hypothesis & directions
2. Experiments & expected results
3. Roadmap

Milestones/how to get there

1. Rewrite the environment(s) ★★☆
   1.1 Code logic for new environment [~1 week]
   1.2 Check everything works as expected (unit testing) [~1 week]
   1.3 Bugs? [~1 week]
2. Baseline training on new environment (convergence, hyperparameter tweaking, etc.) ★★★⚠ [1 week – 1 month]
3. Experiments
   3.1 Task code ★☆☆ [~1 week]
   3.2 Training ★★☆⚠ [~2 week]
   3.3 Analysis code ★★☆ [~2 week]

Milestones/how to get there

1. Rewrite the environment(s) ★★☆
   1.1 Code logic for new environment [~1 week]
   1.2 Check everything works as expected (unit testing) [~1 week]
   1.3 Bugs? [~1 week]
2. Baseline training on new environment (convergence, hyperparameter tweaking, etc.) ★★★⚠ [1 week – 1 month]
3. Experiments
   3.1 Task code ★☆☆ [~1 week]
   3.2 Training ★★☆⚠ [~2 week]
   3.3 Analysis code ★★☆ [~2 week]

Milestones/how to get there

1. Rewrite the environment(s) ★★☆
    1.1 Code logic for new environment [~1 week]
    1.2 Check everything works as expected (unit testing) [~1 week]
    1.3 Bugs? [~1 week]
2. Baseline training on new environment (convergence, hyperparameter tweaking, etc.) ★★★⚠ [1 week – 1 month]
3. Experiments
    3.1 Task code ★☆☆ [~1 week]
    3.2 Training ★★☆⚠ [~2 week]
    3.3 Analysis code ★★☆ [~2 week]

Milestones/how to get there

1. Rewrite the environment(s) ★★☆
   - 1.1 Code logic for new environment [~1 week]
   - 1.2 Check everything works as expected (unit testing) [~1 week]
   - 1.3 Bugs? [~1 week]
2. Baseline training on new environment (convergence, hyperparameter tweaking, etc.) ★★★⚠ [1 week – 1 month]
3. Experiments
   - 3.1 Task code ★☆☆ [~1 week]
   - 3.2 Training ★★☆⚠ [~2 week]
   - 3.3 Analysis code ★★☆ [~2 week]

Milestones/how to get there

1. Rewrite the environment(s) ★★☆
   1.1 Code logic for new environment [~1 week]
   1.2 Check everything works as expected (unit testing) [~1 week]
   1.3 Bugs? [~1 week]
2. Baseline training on new environment (convergence, hyperparameter tweaking, etc.) ★★★⚠ [1 week – 1 month]
3. Experiments
   3.1 Task code ★☆☆ [~1 week]
   3.2 Training ★★☆⚠ [~2 week]
   3.3 Analysis code ★★☆ [~2 week]

Milestones/how to get there

1. Rewrite the environment(s) ★★☆
   1.1 Code logic for new environment [~1 week]
   1.2 Check everything works as expected (unit testing) [~1 week]
   1.3 Bugs? [~1 week]
2. Baseline training on new environment (convergence, hyperparameter tweaking, etc.) ★★★⚠ [1 week – 1 month]
3. Experiments
   3.1 Task code ★☆☆ [~1 week]
   3.2 Training ★★☆⚠ [~2 week]
   3.3 Analysis code ★★☆ [~2 week]

Milestones/how to get there

1. Rewrite the environment(s) ★★☆
   - 1.1 Code logic for new environment [~1 week]
   - 1.2 Check everything works as expected (unit testing) [~1 week]
   - 1.3 Bugs? [~1 week]
2. Baseline training on new environment (convergence, hyperparameter tweaking, etc.) ★★★⚠ [1 week – 1 month]
3. Experiments
   - 3.1 Task code ★☆☆ [~1 week]
   - 3.2 Training ★★☆⚠ [~2 week]
   - 3.3 Analysis code ★★☆ [~2 week]

Milestones/how to get there

1. Rewrite the environment(s) ★★☆
   1.1 Code logic for new environment [~1 week]
   1.2 Check everything works as expected (unit testing) [~1 week]
   1.3 Bugs? [~1 week]
2. Baseline training on new environment (convergence, hyperparameter tweaking, etc.) ★★★⚠ [1 week – 1 month]
3. Experiments
   3.1 Task code ★☆☆ [~1 week]
   3.2 Training ★★☆⚠ [~2 week]
   3.3 Analysis code ★★☆ [~2 week]

Milestones/how to get there

1. Rewrite the environment(s) ★★☆
   1.1 Code logic for new environment [~1 week]
   1.2 Check everything works as expected (unit testing) [~1 week]
   1.3 Bugs? [~1 week]
2. Baseline training on new environment (convergence, hyperparameter tweaking, etc.) ★★★⚠ [1 week – 1 month]
3. Experiments
   3.1 Task code ★☆☆ [~1 week]
   3.2 Training ★★☆⚠ [~2 week]
   3.3 Analysis code ★★☆ [~2 week]

Thanks!