BADGER

Release 0.1

May 27, 2020

Contents

A memory-based multi-agent meta-learning architecture and learning procedure

The Badger architecture allows for the learning of a shared communication policy that enables the emergence of rapid adaptation to new and unseen environments by learning to learn learning algorithms through communication. Behavior, adaptation and learning to adapt emerges from the interactions of homogeneous experts inside a single agent.

The architecture should allow for generalization beyond the level seen in existing methods, in part due to the use of a single policy shared by all experts within the agent as well as the inherent modularity of the architecture.

CHAPTER 1

Contents

1.1 Installation

1.1.1 Installing from source

To install from the source, clone the project with git:

```
git clone https://github.com/JamesPHoughton/pysd.git
```

1.1.2 Required Dependencies

The required dependencies to run the experiments is available in the file requirements.txt

1.2 Experiments

1.2.1 Attention based: Generalization across dimensions

1.2.2 Attention based: Communication through attention

1.2.3 MARL: Scalable reinforcement learning from local observations

1.3 Architecture

The Badger architecture (Rosa, et.al 2019) consists of an agent containing many experts. Each expert operates on the same fixed policy but has unique internal states.

The class .. py:class:: MultitaskAgent defines the architecture of the agent. It consists of the .. py:class:: BadgerAgent and .. py:class:: AttentionLayer classes.

1.3.1 BadgerAgent

The class .. py:class:: BadgerAgent consists of key, value and query networks.

The key network consists of two linear transformations - key1, linear transformation of size HIDDEN + ID to HIDDEN and key2, a linear transformation of size HIDDEN to KEY.

The value network consists of a linear transformation of size HIDDEN + ID to HIDDEN and the query network is a linear transformation from size HIDDEN + ID to KEY.

The valnet network consists of convolutional transformation of size HIDDEN + ID + HIDDEN to HIDDEN and HIDDEN to HIDDEN and another convolutional transformation of size HIDDEN to HIDDEN.

The htoh network consists of a convolutional transformation of size HIDDEN + HIDDEN to HIDDEN.

the htog network consists of a convolutional transformation of size HIDDEN + HIDDEN to HIDDEN.

The BadgerAgent uses the Adam optimizer.

The h0 parameter is of size 1 * HIDDEN * 1.

init_rollout()

This method initializes the rollout parameters hid and static_id.

generate_messages()

The method .. py:method:: generate_messages() returns a key, query and value

key, query and value are different views of the concatenation of the hid and static_id parameters.

receive_attention()

Updates the hid parameter by merging int he hidden state with attention information.

CHAPTER 2

Code

The code for this package is available at: https://github.com/GoodAI/badger-2019

CHAPTER 3

Indices and tables

Index