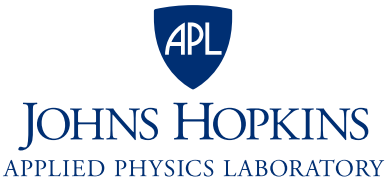

JHU/APL
http://www.jhuapl.edu/

Title:
Lifelong Learning Machines
Preliminary Metrics Documentation

Project Participants:
JHU/APL Test & Evaluation Team

Authors:
Megan Baker
Gautam Vallabha

Date:
November 22, 2019

Contents

Chapter 1

Background

In this document we introduce metrics for reinforcement learning tasks which...

1.1 Core Capabilities

We rely on the BAA to define the goal of the program and the metrics by which we can assess whether an algorith exhibits Lifelong Learning. Thus, we invite our audience to review the five core capabilities and their definitions below.
1. Continual Learning
2. Adapting to New Tasks
3. Selective Plasticity
4. Goal-Driven Perception
5. Safety

Chapter 2

System Architecture

Syllabus is run, logs are generated, Metrics code is run.

Metrics code scrapes the logs, extracts the relevant information and produces a set of numbers which are used to evaluated each Core Capability

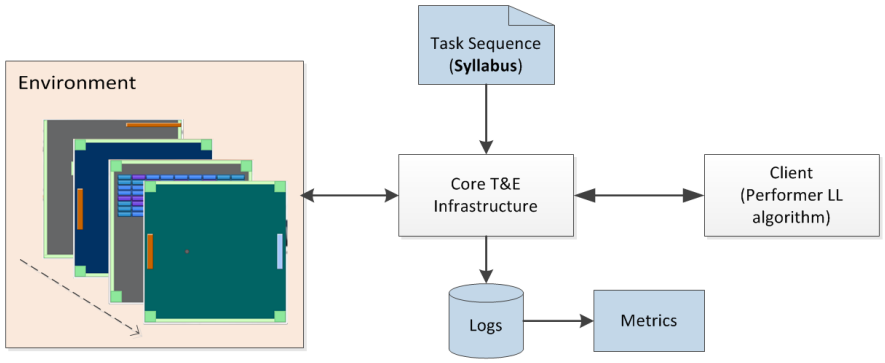

Figure 2.1: This figure describes the production of metrics from log files in the Test and Evaluation Framework.

Chapter 3

Syllabus Types

With the Core Capabilties in mind, we move on to intentional design of the syllabi. We remind our audience that the calculation of L2M metrics seeks to answer a question that may differ from traditional reinforcement learning evaluation criteria, and thus, we enforce a rigid structure to evaluation tasks.

3.1 Term Definitions

- Phase - A phase is - Phase number should be incremented, starting with phase 1 - Phase type can be either "train" or "test"
- Block - A block is a unique combination of task and parameters. It is a subcomponent of phase that is automatically assigned by the logging code and does not need to be annotated by the syllabus designer.
- Task - ADD DEFINITION FROM RELEASED DOCUMENTATION HERE
- Episode - ADD DEFINITION FROM RELEASED DOCUMENTATION HERE

3.2 Syllabus Types by Core Capability

Continual Learning

1) This syllabus consists of a single task with parametric variations (P1, P2, P3, ...)
Can involve interpolation of parameters, noisy parameters, etc. but has only one task
2) Each phase consists of a training block and an optional evaluation block
Training block is training on one or more parametric variations
Evaluation block allows Type II metrics to be computed

Question: Can the agent adjust to changes in the environment?
Metric: Recovery time (relative to previously trained parameter baseline)
Calculated within and across training blocks

After variation is introduced, does performance fall? Does it recover to the previous level? How quickly does it recover?

Question: Can the agent maintain performance on previously learned parameters after being trained with new ones?
Metric: Performance on previously learned parameters
Recommended to be calculated within evaluation block

Adaptation to New Tasks

This syllabus consists of multiple, distinct tasks (T1, T2, T3, ...). Parametric variations within a task are only exercised in Subtype C
Each phase consists of a training block and an evaluation block
Training block is training on one task at a time, and can be over some parametric variation, but only in Subtype C
Evaluation block allows Type B metrics to be computed

Subtype A: No expectation of skill transfer
Parametric variations within a task are not exercised.
Question: Can the agent learn a new task without forgetting the old task?
Metric: Recovery time (relative to previous task baseline)
Calculated within and across training blocks
After a new task is introduced, does performance fall? Does it recover to the previous level? How quickly does it recover?
Metric: Performance (reward statistics) on previously learned tasks
Calculated within evaluation blocks
Note that this is not to assess forward or reverse transfer; transfer is assessed in ANT Subtype B

Subtype B: Assess basic skill transfer
Parametric variations within a task are not exercised.
Question: Does the agent transfer knowledge from a learned task to a new task?
Metric: Recovery time (relative to previous task baseline)
Calculated within and across training blocks
After a new task is introduced, does performance fall? Does it recover to the previous level? How quickly does it recover?
Metric: Performance (reward statistics) on previously learned tasks
Calculated within evaluation blocks
Note that this is not to assess forward or reverse transfer; transfer is assessed in ANT Subtype B
Metrics are:
Time (number of episodes) to achieve threshold performance on Task 2 after being trained

on Task 1
Transfer matrix

Subtype C: with Parametric skill transfer
Syllabus has: Parametric variation in Task 1 and then see if that transfers to parametric variation in task 2.
L2Arcade Example: vary paddle width in Pong; does this transfer to varying paddle width in Breakout?
this can be formulated in both a forward and reverse transfer.
Questions
Transfer of general ability: does parametric variation in Pong improve Breakout time-to-learn at all
Transfer of continual learning: Does parametric variation (continual learning) in Pong improve continual learning in Breakout?
Metrics are:
Amount of time required to learn vs from scratch

3.3 Metric Definitions

1. Saturation Value - Purpose: The saturation value is computed to quantify the maximum maintained value by the agent.
- Calculated by: Since multiple rewards may be logged per episode, the mean is taken per episode. Then, the max of the rolling average is taken of the mean reward per episode with a smoothing parameter, s (default, 0.1)
- Compared to: Future or single task expert saturation values of the same task
- Computed for: CL, ANT_A, ANT_B

2. Time to Saturation - Purpose: Time to saturation is used to quantify how quickly, in number of episodes, the agent took to achieve the saturation value computed above.
- Calculated by: The first time the saturation value (or above) is seen, that episode number is recorded
- Compared to: Future times to saturation of the same task
- Computed for: CL, ANT_A, ANT_B

3. Normalized Integral of Reward/Time - Purpose: Taking the Integral of Accumulated Reward over Time allows for a more robust comparison of the time to learn a particular task, taking into account both the shape and saturation of the learning for future comparison. Has limitations; must be normalized by length; only training phases can be compared to each other
- Calculated by: Integrating reward over time, then dividing by the number of episodes used

to accumulate the reward
- Compared to: Future training instances of this metric
- Computed for: ANT_B

4. Recovery Time - Purpose: Recovery time is calculated to determine how quickly (if at all) an agent can "bounce back" after a change is introduced to its environment
- Calculated by: After some training phase achieves a saturation value, determine how many episodes, if any, it takes for the agent to regain the same performance on the same task
- Compared to: An agent's recovery time is comparable across tasks
- Computed for: CL

5. Performance Maintenance on Test Sets
- Purpose: Performance maintenance on test sets is calculated to determine whether an agent catastrophically forgets a previously learned task
- Calculated by: Comparing all computed metrics on the train set (saturation values, time to saturation, etc) on the test set and computing the difference in performance
- Compared to: N/A
- Computed for: CL, ANT_A, ANT_B

6. Performance relative to STE (training) - Saturation Value, Time, Integral
- Purpose: STE Relative performance assesses whether a lifelong learner outperforms a traditional learner.
- Calculated by: Normalizing metrics computed on the lifelong learner by the same metrics computed on the traditional learner
- Compared to: N/A
- Computed for: CL, ANT_A, ANT_B

7. Forward Transfer (cross tasks) - Purpose: Forward transfer assesses whether an agent transfers knowledge from a learned task to a new task
- Calculated by:
- Compared to: STE performance
- Computed for: CL, ANT_A, ANT_B

8. Forward Transfer (cross tasks) - Purpose: Forward transfer assesses whether an agent transfers knowledge from a learned task to a new task
- Calculated by:
- Compared to: STE performance
- Computed for: CL, ANT_A, ANT_B

9. Time to Learn (cross tasks) - Purpose: Time to learn (cross tasks) assesses whether parametric variation in learning task 1 improve time to learn in task 2

- Calculated by:
- Compared to:
- Computed for: ANT-C

10. Time to Continually Learn (cross tasks) - Purpose: Time to continually learn (cross tasks) assesses whether learning task 1
- Calculated by:
- Compared to:
- Computed for: ANT-C

3.4 Example Syllabi

Put up some JSON files here

Chapter 4

Metrics Code

4.1 Code Release

Metrics code will be released to GitHub!

4.2 How to Use

Assumptions and Requirements

Without further ado,

1. Log files **must** include logged reward, and the column in the data file **must** be named reward. Without this column, the metrics code will fail. This is assumed to be logged per episode.

2. Single Task Expert Saturation values for each task **must** be included in a JSON file found in $L2DATA/taskinfo/info.json and without this file, the metric "Comparison to STE" cannot be calculated. Further, the task names contained in the JSON file must match the names in the log files exactly. The format for this file will be:

"task_name_1" : 0.8746, "task_name_2" : 0.9315, ..., "task_name_n" : 0.8089

3. Syllabi used to generate the log files **must** include annotations with phase information and shall conform to the following convention:

Phase annotation format:
"$phase": "1.train", "$phase": "1.test", "$phase": "2.train", "$phase": "2.test", etc

Structure:

+ CL

Consists only of a single task with parametric variations exercised throughout the syllabus. Testing phase is optional, but recommended. The purpose of this type of syllabus is to

assess whether the agent can adjust to changes in the environment and maintain performance on the previous parameters when new ones are introduced

+ ANT, subtype A

Consists of multiple tasks with no parametric variations exercised throughout the syllabus. Testing phase is mandatory. The purpose of this type of syllabus is to assess whether the agent can learn a new task without forgetting the old task and does not seek to assess knowledge transfer.

+ ANT, subtype B

Consists of multiple tasks with no parametric variations exercised throughout the syllabus. Testing phase is mandatory. The purpose of this type of syllabus is to assess whether the agent can transfer knowledge from a learned task to a new task.

+ ANT, subtype C

Consists of multiple tasks with parametric variations exercised throughout the syllabus. Testing phase is mandatory. The purpose of this type of syllabus is to assess whether the agent can transfer knowledge from a learned task with parametric variation to a new task with parametric variation.

Write your Own Custom Metric

The steps required to run your new metric in the existing metrics pipeline are as follows:

1. Write your custom metric, MyCustomMetric in agent.py according to the structure set in l2metrics/core.py

More details regarding the structure of the AgentMetrics class can be found in l2metrics/core.py - your metric code to actually compute a custom metric goes in the required calculate method. Note that the required arguments are the log data, the phase_info, and the metrics_dict. The metrics dict is filled by each metric in its turn, whereas the log data and the phase info are extracted in the AgentMetricsReport constructor from the logs via two helper functions

l2metrics/util.py - (read_log_data): scrapes the logs and returns a pandas dataframe of the logs and task parameters l2metrics/_localutil.py - (parse_blocks): builds a pandas dataframe of the phase information contained in the log data

2. Insert MyCustomMetric to be added into the appropriate default metric lists for any syllabus type you want the metric to be calculated for - CL, ANT_A, and/or ANT_B

The AgentMetricsReport adds the default list of metrics based on the passed syllabus type (a mandatory parameter to run the metrics code via command line). AgentMetricsReport will only add your new metric to the default list if you insert it in the _add_default_metrics method for your desired syllabus type

3. Run l2metrics/__ main __.py with the appropriate command line parameters and watch your metric be reported when you call the report method