Cpts 570 Homework 4

Xinyu Liu

December 2021

1

Epsilon: 0.1, Iteration: 1247648

Epsilon: 0.2, Iteration: 296917

Epsilon: 0.3, Iteration: 110857

Temperature: 100.0, Iteration: 46476

Temperature: 10, Iteration: 42360

Temperature: 1, Iteration: 25702

Temperature = temperature * 0.9 for each iteration

Iterations can be improved if I put a threshold on value diff instead of completely equal.

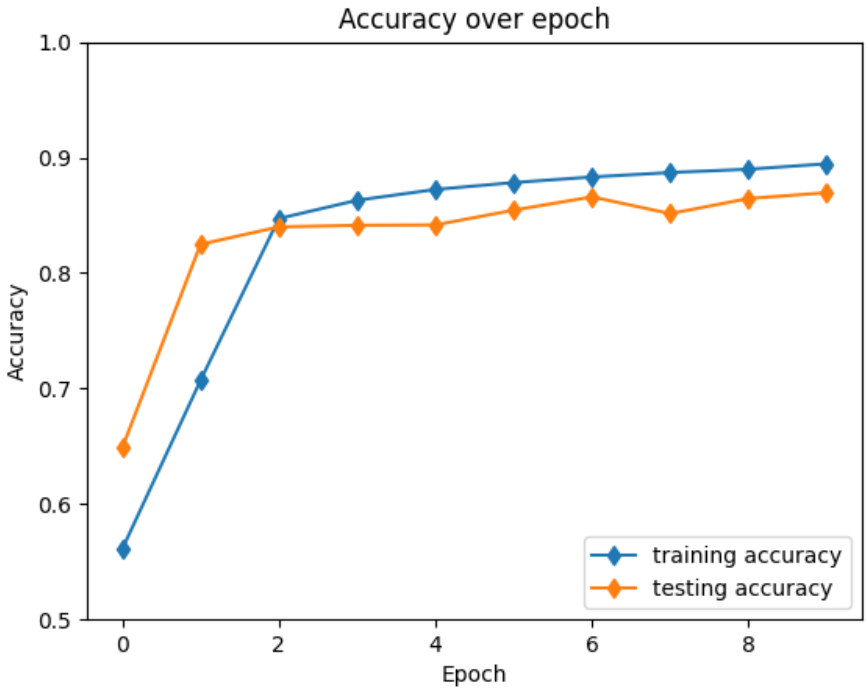
Q-values for the path:

```
========================= Path ============================

[0, 0], CELL, reward:0,  Q-values: {'up': 0, 'down': 0.18530201888516112, 'left': 0, 'right': 0.18530201888516112}
[0, 1], CELL, reward:0,  Q-values: {'up': 0.16677181699664362, 'down': 0.16677181699664362, 'left': 0, 'right': 0.205891132094625}
[1, 1], CELL, reward:0,  Q-values: {'up': 0.18530201888516112, 'down': 0, 'left': 0.18530201888516112, 'right': 0.22876792454958483}
[2, 1], CELL, reward:0,  Q-values: {'up': 0.205891132094625, 'down': 0, 'left': 0.205891132094625, 'right': 0.25418658283287354}
[3, 1], CELL, reward:0,  Q-values: {'up': 0.22876792454958483, 'down': 0, 'left': 0.22876792454958483, 'right': 0.2824295364809737}
[4, 1], CELL, reward:0,  Q-values: {'up': 0.25418658283287354, 'down': 0, 'left': 0.25418658283287354, 'right': 0.31381059608997386}
[5, 1], CELL, reward:0,  Q-values: {'up': 0.2824295364809737, 'down': 0.34867844009997395, 'left': 0.2824295364809737, 'right': 0.2824295364809737}
[5, 2], CELL, reward:0,  Q-values: {'up': 0.31381059608997386, 'down': 0.38742048899997406, 'left': 0, 'right': 0}
[5, 3], CELL, reward:0,  Q-values: {'up': 0.34867844009997395, 'down': 0.0, 'left': 0, 'right': 0.43046720999997423}
[6, 3], CELL, reward:0,  Q-values: {'up': 0, 'down': 0.0, 'left': 0.38742048899997406, 'right': 0.4782968999999744}
[7, 3], CELL, reward:0,  Q-values: {'up': 0, 'down': 0.5314409999999745, 'left': 0.43046720999997423, 'right': 0.43046720999997423}
[7, 4], CELL, reward:0,  Q-values: {'up': 0.4782968999999744, 'down': 0.5904899999999778, 'left': 0.0, 'right': 0.4782968999999744}
[7, 5], CELL, reward:0,  Q-values: {'up': 0.5314409999999745, 'down': 0.6560999999999813, 'left': 0.0, 'right': 0.0}
[7, 6], CELL, reward:0,  Q-values: {'up': 0.5904899999999778, 'down': 0.5904899999999731, 'left': 0.7289999999999852, 'right': 0.0}
[6, 6], CELL, reward:0,  Q-values: {'up': 0.0, 'down': 0.0, 'left': 0.8099999999999896, 'right': 0.6560999999999862}
[5, 6], CELL, reward:0,  Q-values: {'up': 0.8999999999999946, 'down': 0.0, 'left': 0, 'right': 0.7289999301615402}
[5, 5], CELL, reward:1,  Q-values: {'up': 0, 'down': 0, 'left': 0, 'right': 0}
 | > | v | > | > | v | v | v | > | > | v |
 | > | > | > | > | > | v | < | > | > | v |
 | ^ | X | X | X | X | v | X | X | X | v |
 | ^ | < | < | - | X | > | > | v | v | < |
 | ^ | < | < | < | X | - | - | v | < | < |
 | ^ | < | < | < | X | G | - | v | - | ^ |
 | ^ | < | < | < | X | ^ | < | < | - | v |
 | ^ | ^ | ^ | - | X | - | - | ^ | < | < |
 | ^ | ^ | ^ | > | > | > | > | ^ | < | < |
 | ^ | < | ^ | > | > | > | > | ^ | ^ | ^ |
```

Q-values for each state-action pair can be found in code/ouput/output.txt

2



3

Although developing a ML system is fast and easy, the cost of maintaining it is high. This cost is also known as technical debt. Following is the list of debt that author talked about:

1. Erode Boundaries

Encapsulation and modular design cannot be enforced on ML. There are 3 ways to erode the supposed boundary: **Entanglement**, **Correction cascades**, **Undeclared consumers**.

2. Data Dependencies

When building a ML system, data dependency could be heavier than code dependency, and detection of data dependency is hard.

3. Feedback Loops

ML systems influence their own behavior over time. It's hard to predict a behavior before the model is computed. Gradually, these feedback loops become hard to detect.

4. Anti Patterns

Only a small portion of code is used for learning and prediction, majority of the code is about "plumbing", which means interconnection components that are not doing ML related work.

5. Configuration Debt

There are also a lot of code for configuration. Before running any algorithm, there is a good portion of code for data processing and feature selection/reduction. There's also hyperparameter tuning for the algorithm. All these parts of the code are also considered as technical debts.

6. Changes in the External World

Machine learning is about studying on external world, however, the external world is dynamic. Things like threshold, bias, limits are constantly changing.

There are also some other possible technical debts that are uncovered. In conclusion, there needs to be a way to lower the cost of maintaining ML systems.

4

This article pretty much complements the one above. It provides a set of tests such that a ML system maximizes production-readiness and minimizes technical debt. There are total of 28 tests. They can split into the following structure:

1. Features and data
    - feature distribution, feature benefits, feature costs, feature policies, data privacy, new feature time, test feature creation
2. Model development
    - model code review, offline online metrics, tune hyperparameters, model staleness, baseline model, quality on slices, inclusiveness
3. ML infrastructure
    - training reproducibility, unit test model, integration test, quality before serving, debugger, canary before serving, serving model rollback
4. Production monitoring
    - data dependencies, data invariants, training serving skew, model staleness, numerical instability, performance regression, quality regression