Autonomous Management of Energy-Harvesting IoT Nodes Using Deep Reinforcement Learning

IEEE SASO 2019

ROBIOT: Reinforcement Learning for Intelligent, Autonomous IoT Devices

饮水机 Alexander Kraemer: NTNU IoT Lab

Kerstin Bach: NTNU AI Lab

Gavin Taylor: USNA

Abdulmajid Murad
Why RL in IoT?

IoT Node

Super Mario

Real Environment

Super Mario Environment
How?

IoT Node

Real Environment

Super Mario

Super Mario Environment

Open AI Gym

?
How?

IoT Node

Real Environment

Sensor Gym

Open AI Gym

Super Mario

Super Mario Environment
Which Application?

Sensor Gym

Device Management

$S_t = ?$

$R_t = ?$

$\alpha_t = ?$

$\pi \theta = ?$

IoT Node

Data Collection

Deployment (Policy Update)
Power Management Energy-Harvesting IoT

\[ S_t = \text{Energy buffer} \]

\[ = \text{Harvested energy} \]

\[ = \text{Weather forecast} \]

\[ = \text{(other non-important states)} \]

\[ R_t = ? \]

\[ \alpha_t = \text{How much energy to spend} \]

\[ \approx \text{(approximated by duty cycle)} \]

\[ \pi \theta = ? \]
Reward function Based on Energy Neutrality

\[ R_E(s_t) = \begin{cases} 
500 & \text{if } Edist_t = 0 \text{ W h} \\
500 - \frac{Edist_t}{10} & \text{if } 0 \text{ W h} < Edist_t \leq 1 \text{ W h} \\
10 - \frac{Edist_t}{100} & \text{if } 1 \text{ W h} < Edist_t \leq 5 \text{ W h} \\
-500 & \text{if } 5 \text{ W h} < Edist_t
\]
Which RL method?

Sensor Gym

$S_t =$
- Energy buffer
- Harvested energy
- Weather forecast
- (& other non-important states)

$\tau_t =$
Distance from
Energy Neutrality

$\alpha_t =$
How much energy to spend
(approximated by duty cycle)

$\pi \theta =$
Policy Gradient (e.g. PPO)
(Continuous states and actions)

Value-function (e.g. SARSA)
(Discrete states and actions)
Sensor Gym Simulation

Sensor specification based on a realistic energy-harvesting IoT node (scaled up version up of a TMote Sky node)

Simulate the harvested energy by calculating the energy generated by a 6W solar panel using solar radiation data.

Used weather data of Tokyo and Wakkanai for 2010 & 2011.

Trained on data of Tokyo for 2010 and tested on 2011 (both cities)
Performance Results

(One week sample of Tokyo 2011)
Performance Results

(One week sample of Tokyo 2011)
Performance Results

- Linear Programming Optimization Method:
  \[ RMS \text{ Edist}_{\text{week}} = 3.69\% \]

- Agent Trained with PPO Algorithm:
  \[ RMS \text{ Edist}_{\text{week}} = 4.97\% \]

- Agent Trained with SARSA(\(\lambda\)) Algorithm:
  \[ RMS \text{ Edist}_{\text{week}} = 5.51\% \]

(One week sample of Wakkanai 2011)
Performance Results

One week sample of Wakkanai 2011

- Linear Programming Optimization Method
  \( RMS \text{ Edist}_{\text{week}} = 3.19\% \)

- Agent Trained with PPO Algorithm
  \( RMS \text{ Edist}_{\text{week}} = 3.85\% \)

- Agent Trained with SARSA(\( \lambda \)) Algorithm
  \( RMS \text{ Edist}_{\text{week}} = 4.41\% \)
Performance Results
(Over the whole year of 2011)
We trained 20 agents with PPO and 20 with SARSA with data of 2010 and evaluated their policies on data of 2011.
What is Wrong with a Reward Function based on Energy Neutrality?
The duty cycles of both RL policies are subject to high variance.

From an RL point-of-view, this maximizes the reward.

It is not a behavior appropriate for IoT nodes.

Typically, we want to cover a phenomenon as continuously as possible, and...
Reward based on Energy doesn’t Reflect the True Application Goals
Reward function Based on Energy Neutrality

Objectives?

游戏操作符 We want the agent to maximize the sum of the duty cycle over time:

\[ G_D = \sum_{t=0}^{\Gamma} D_t \]

游戏操作符 We want to minimize the occurrence of failures over time:

\[ G_F = \sum_{t=0}^{\Gamma} \begin{cases} 1 & \text{if } B_t = 0 \\ 0 & \text{if } B_t > 0 \end{cases} \]

游戏操作符 We want to minimize the variance of the selected duty cycles

\[ G_{Var} = \sum_{t=0}^{\Gamma} Var_t, \quad Var_t = |D_t - D_{t-1}| \]

Combined conflicting objectives into one reward function:

\[
R_A(s_T) = \sum_{t=1}^{T} \begin{cases} 
D_t - \zeta [Var_t]^2 & \text{if } B_t > 0 \\
-\bar{F} & \text{if } B_t = 0 
\end{cases}
\]
Performance Results

We trained more than 300 agents to explore the influence of the designed reward function and training hyperparameters on the agent behavior.
Hyper-parameters Tunning

- Damping factor (ζ), learning rate (α), batch size, discount factor (γ) and trace decay parameter (λ) of the advantage function.
Performance Results

Four selected agents trained with different damping factors ($\zeta$) in their reward function.

| $\zeta$  | Utilized Energy ($M_D\%$) | Variance Mean ($M_{Var}$) | Power Failures ($M_f$) |
|----------|--------------------------|---------------------------|----------------------|
| 0.1      | 77                       | 11                        | 2.5                  |
| 0.05     | 94                       | 14                        | 9.7                  |
| 0.01     | 95                       | 17                        | 12                   |
| 0.001    | 98                       | 23                        | 25.6                 |
Performance Results

Running four selected agents over the whole year.
Results show that shallower networks perform as well as deeper networks.

An agent’s policy can be approximated with a neural network that requires low computational effort and memory footprint.
Master Student (Aksel Vincent Berg) implemented NN in Resource-Constrained Devices (nRF52840 Microcontroller)
Thank you!