# A Neural Network-Based Proposal for Optimizing Battery Life and Autonomous Task Management in Nanosatellites

Carlos Padilla enigmak9@protonmail.com

ICN, UNAM

April 22, 2024

## **Table of Contents**

Optimizing Battery Life in NanoSatellites

Carlos Padilla

#### LIN

Energy Efficient Requirements

Requirements

Learning

DQN

Bellman's equation

equation
Task Execution

Model

Definition

vectors

Implementation

Implementation

1 LINX

Energy Efficiency Requirements

Reinforcement Learning

DQN

Bellman's equation

6 Explanation of the Bellman equation

Task Execution Model

8 Definitions

Status and vectors

Expected output vector

Implementations

Architecture

### LINX

Optimizing Battery Life in NanoSatellites

Carlos Padilla

LINX

Energy Efficier Requirements

Reinforcemei Learning

\_\_\_\_\_

DQI

equation Explanatio

the Bellman equation

Model

Definitions

Status and vectors

Expected out vector

Implementation

Architecture



## **Energy Efficiency Requirements**

Optimizing Battery Life in NanoSatellites

Carlos Padilla

LIN)

Energy Efficiency Requirements

Reinforcemer Learning

Bellman's

Explanation of the Bellman equation

Task Execution Model

Definitions

Status and vectors

Expected outpose vector

Implementatio

Architecture

The network must ensure that the battery's state of charge (SoC) remains above 20%, avoiding long periods below 30% to ensure the satellite's longevity and effectiveness.

# Reinforcement Learning

Optimizing Battery Life in NanoSatellites

Carlos Padilla

LIN)

Energy Efficien Requirements

Reinforcement Learning

DQN

Bellman's equation

Explanation of the Bellman equation

Definitions

Status and

Expected out

Implementatio

Architecture

Reinforcement learning is an area of machine learning inspired by behavioral psychology, which focuses on determining what actions a software agent should choose in a given environment to maximize some notion of cumulative reward or prize.

## DQN

Optimizing Battery Life in NanoSatellites |

Carlos Padilla

DON

Our model will be a feedforward neural network that takes the difference between the current and previous screen patches. It has two outputs, representing Q(s, left) and Q(s, right) (where s is the network input). Effectively, the network tries to predict the expected return of performing each action given the current input. Reinforcement Learning (DQN) Tutorial

## Bellman's equation

Optimizing Battery Life in NanoSatellites

Carlos Padilla

LIN)

Energy Efficien Requirements

Reinforcemer

Learning

Dellarada

Bellman's equation

Explanation of the Bellman equation

Model

Definitions

Status and vectors

Expected out

Implementatio

Architecture

$$V(s) = \max_{a} \left\{ R(s, a) + \gamma V(s') \right\} \tag{1}$$

## Explanation of the Bellman Equation

Optimizing Battery Life in NanoSatellite

Carlos Padilla

LIN

Energy Efficier Requirements

Reinforcemer Learning

DQN

Explanation of

equation
Task Executio

Definitions

Status and vectors

Implementation

mplementation:

The Bellman equation is defined as:

$$V(s) = \max_{a} \left\{ R(s, a) + \gamma V(s') \right\}$$
 (2)

#### Where:

- V(s) is the value of state s. It represents the maximum expected return when the agent is in state s.
- max<sub>a</sub> denotes that we are selecting the action a that maximizes the following value.
- R(s, a) is the immediate reward received for taking action a in state s.
- $\gamma$  is the discount factor, balancing the importance of immediate rewards versus future rewards. It ranges from 0 to 1.
- V(s') is the value of the subsequent state s', representing the expected return from that new state.



Optimizing Battery Life in NanoSatellites

Carios Fauili

LIN

Energy Efficier Requirements

Reinforceme

Learning

DQN

Explanation of the Bellman equation

Task Execution

**Definitions** 

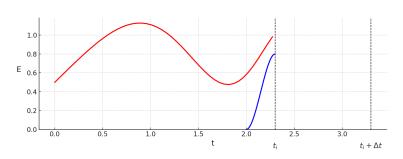
Status and

Expected ou vector

Implementation

A section and the

 $W_0$ 



## Task Execution Model

Optimizing Battery Life in NanoSatellites

Carlos Padilla

LIN

Energy Efficient Requirements

Reinforcem Learning

DON

Bellman's equation

equation
Task Execution

Model

Definitions

Status and vectors

Expected outp

Implementatio

Architecture

$$\mathsf{M}_{j} = \left( \left[ I, t_{j}^{\prime}, \Delta t_{j}, P_{j}^{R}, P_{j}^{D} \right], \left[ W_{I}^{R}, W_{I}^{D} \right] \right)$$

## **Definitions**

Optimizing Battery Life in NanoSatellites

Carlos Padilla

LIN

Energy Efficient Requirements

Reinforceme Learning

Learning

Bellman's equation

Explanation of the Bellman equation

Model

Definitions

vectors

vector

Implementatio

Architecture

#### Where:

- I Initial time.
- $t'_i$  Time at which task j is scheduled to be executed.
- $\Delta t_j$  Time duration for which task j is executed.
- P<sub>j</sub><sup>R</sup> Execution priority: indicates the relative importance of task j compared to other tasks in terms of when they should be executed.
- P<sub>j</sub><sup>D</sup> Data download priority: indicates the relative importance of task j in terms of when associated data should be downloaded.
- $[W_i^R, W_i^D]$  Intrinsic properties.



## Status and Vectors

Optimizing Battery Life in NanoSatellites

Carlos Padilla

LIN

Energy Efficien Requirements

Reinforcemo

Learning

DQN

Bellman's equation

equation
Task Execution

Definitions

Status and vectors

vector

Architecture

#### Status information:

$$E(t_i); \quad \frac{dE}{dt}; \quad \frac{d^2E^-}{dt^2}$$

$$W(t_i); \quad \frac{dW^-}{dt}; \quad \frac{d^2W^-}{dt^2}$$

### Data vector for project:

$$D = (d_1, d_2, \dots, d_J)$$
 amount of data of task  $j$ 

#### **Execution status vector:**

$$S = (s_1, s_2, \dots, s_J)$$
 /  $s_j = \begin{cases} 0 & \text{if } j \text{ not executed} \\ 1 & \text{if } j \text{ is executed} \end{cases}$ 

# Expected output vector

Optimizing Battery Life in NanoSatellites

Carios Padil

LINX

Energy Efficier Requirements

Reinforceme Learning

DON

Bellman's

equation

Explanation of the Bellman equation

Model Model

Definitions

Status and vectors

Expected output vector

Implementation

Arabitaatura

-	1	2	3	4	5	
<i>t</i> 1	0	0	1	0	0	• • • •
t2	1	0	0	0	0	
t3	0	1	0	0	0	
t4	0	0	0	0	1	
<i>t</i> 5	0	0	0	1	0	
		:			:	٠



## **Implementations**

Optimizing Battery Life in NanoSatellites

Carlos Padilla

LIN)

Requirements

Reinforceme Learning

DON

DQN

Bellmar equation

Explanation of the Bellman equation

Model

Definitions

Status and vectors

Expected outp

Implementations

Architecture

Red Plus 001 –jorge vega Red 002 –jorge vega Red 003 –jorge vega DQN (12 march 2024) – Carlos Retask priorities –Jorge Vega SoC Cycles prediction –Jorge Vega Power Graph (09 april 2024) –ciph

## **Architecture**

Optimizing Battery Life in NanoSatellites

Carlos Padilla

LINX

Energy Efficier Requirements

Reinforceme

Learning

Dellesses

equation

Task Execution

**Definitions** 

Status and vectors

Expected out

Implementation

Architecture

#### DQN Architecture with Annotations for Neuron Counts



