**Instituto Tecnológico de Aeronáutica - ITA**
**Inteligência Artificial para Robótica Móvel - CT213**
**Alunos**: João Lucas Rocha Rolim, Samir Nunes da Silva e Samuel Afonso de Souza Cavalcante

# Manual do Usuário - Resolução do Jogo de Blackjack via Deep Q-Learning no Ambiente Gymnasium - Exame de CT-213

## 1 Instalação das Dependências

Dividiremos a instalação das dependências em duas subsessões, a depender do sistema operacional utilizado: Windows ou Linux.

### 1.1 Windows

Para criar um ambiente utilizando o Anaconda com as dependências corretas para a execução dos códigos, utilize o comando no *prompt* do Anaconda:

```
conda env create -name <env-name> -file windows-requirements.yml
```

No qual deve-se substuir `<env-name>` pelo nome do novo ambiente. Se o processo for corretamente executado, os seguintes textos aparecerão no *prompt*, seguidos da instalação das bibliotecas:

```
Collecting package metadata (repodata.json):  done
Solving environment:  done

Downloading and Extracting Packages

Preparing transaction:  done
Verifying transaction:  done
Executing transaction:  done
Installing pip dependencies:
```

Ao fim do processo, aparecerão instruções para ativação do ambiente criado:

```
done
#
# To activate this environment, use
#
#    $ conda activate <env-name>
#
# To deactivate an active environment, use
#
```

```
#    $ conda deactivate
```

Com isso, o ambiente em Windows estará pronto para rodar corretamente os arquivos `.py` do projeto.

### 1.2 Linux

Praticamente todo o processo para Linux é análogo. A única diferença ocorre na primeira linha no *prompt* do Anaconda, na qual se deve digitar o seguinte comando:

```
conda env create -name <env-name> -file linux-requirements.yml
```

Com isso, o ambiente em Linux estará pronto para rodar corretamente os arquivos `.py` do projeto.

## 2 Utilização dos *Scripts* em Python do Projeto

Seguem as instruções sobre os *scripts* do projeto:

- Para **treinar** o agente DQN no jogo de *Blackjack*, rode o *script* `train_dqn.py`;
- Para **mudar os hiperparâmetros** do treinamento, edite o dicionário no arquivo `hyperparameters.py`;
- Para **avaliar o agente DQN treinado**, rode o *script* `evaluate_dqn.py`;
- Para **avaliar um agente com política aleatória** como comparação, rode o *script* `evaluate_random.py`.
- O arquivo blackjack.h5 guarda o **agente treinado**. Nos arquivos originais, foi enviado o agente treinado que gerou os resultados presentes no relatório.