# 公立はこだて未来大学 2016 年度 システム情報科学実習
# グループ報告書

**Future University Hakodate 2016 System Information Science Practice**
**Group Report**

## プロジェクト名

AI するディープラーニング

## Project Name

AI love Deep Learning

## グループ名

TORCS Deep Learning

## Group Name

TORCS Deep Learning

**プロジェクト番号/Project No.**

14-B

## プロジェクトリーダ/Project Leader

1014041　福田大知　Daichi Fukuda

## グループリーダ/Group Leader

1014053　高橋将文　Masafumi Takahashi

## グループメンバ/Group Member

1014018　能登楓　Kaede Noto
1014023　伊藤空　Sora Ito
1014053　高橋将文　Masafumi Takahashi
1014066　加藤雅崇　Masataka Kato
1014094　齊藤伶奈　Reina Saito
1014126　源智也　Tomoya Minamoto

## 指導教員

竹之内高志　永野清仁　寺沢憲吾　片桐恭弘

## Advisor

Takashi Takenouchi　Kiyohito Nagano　Kengo Terasawa　Yasuhiro Katagiri

## 提出日

2017 年 1 月 18 日

## Date of Submission

January 18, 2017

# 概要

近年, 様々な分野において人間を模倣できる人工知能が台頭してきている. 人工知能は機械学習などを用いて, 人間が自然に行っている学習能力と同様の知能をコンピュータで実現しようとする技術・手法のことである. 特に, ディープラーニング (深層学習) は画像処理分野で優秀な成果を出している手法である. 本プロジェクトの目標はこれらの機械学習手法を用いて, 人間の思考を模倣・超越することである. 目標についてディスカッションをし, プロジェクトをグループ A, グループ B にわけた. グループ A では配球予想システムの開発, グループ B では人間が操作するよりも, 速く走行できるカーエージェントの開発を目標とした. 本グループの目標は, 人間が操作するよりも速く走行可能なカーエージェントの開発である. カーエージェントの開発には深層強化学習を用いた. 深層強化学習とは強化学習の関数近似に深層学習の技術を適用した技術である. 適切な学習をするための課題は適切なネットワークや報酬, 環境の設定などである. 前期では, Python を用いた Long Short Term Memory(LSTM) の実装, Unity でレースゲームの報酬や環境の構築を行った. 後期では, Asynchronous method, Deep Q-Network(DQN), LSTM などの手法を用いて, オープンソースのカーシミュレータである TORCS という学習環境上でカーエージェントに学習をさせた. Asynchronous method を用いたカーエージェントはオーバルトラックにおいて人間が操作するよりも速く走行することができた.

**キーワード**　人工知能, 機械学習, 深層学習, ディープラーニング, 強化学習, Unity, TORCS

（※文責: 能登楓）

# Abstract

Recently, AI attracts attention because it can imitate humans in various cases. AI is a kind of technology of Machine Learning. We use it to implement some intelligence. These are same intelligence as human's natural learning abilities. Especially, Deep Learning has archived good results in the field of image processing. In this project, our goal is to imitate and surpass human's thoughts with using Machine Learning. After we discussed our goal, we made two groups. These were group A and group B. The members in group A aimed to develop a combination of pitches expectation system. And the members in group B aimed to develop a car agent that can drive cars faster than humans. We belong to group B and use Deep Reinforcement Learning to develop such a car agent. Deep Reinforcement Learning is technique that applies technique of Deep Learning to function approximation of Reinforcement Learning. The problems to learn in good order are to set appropriate network, rewards, and environment. In the first semester, we implemented Long Short Term Memory (LSTM) on Python, rewards, and environment of a racing game on Unity. And we had the car agent learn with using these works. In the second semester, we had the car agent learn with using techniques of Asynchronous method, Deep Q-Network (DQN), and LSTM on TORCS. TORCS is open source car simulator. Finally, Asynchronous method car agent can drive cars faster in an oval track than humans.

**Keyword** Artificial Intelligence, Machine Learning, Deep Learning, Reinforcement Learning, Unity, TORCS

（※文責: 伊藤空）

# 目次


**第 1 章　はじめに** **1**

1.1　背景 . . . . . 1
1.2　目的 . . . . . 1
1.3　従来例 . . . . . 1
1.4　従来の問題点 . . . . . 2
1.5　課題 . . . . . 2

**第 2 章　プロジェクト学習の概要** **3**

2.1　到達目標 . . . . . 3
2.2　問題の設定 . . . . . 3
2.3　課題の設定 . . . . . 3
2.3.1　前期の課題の概要 . . . . . 3
2.3.2　前期の課題 . . . . . 4
2.3.3　前期の反省 . . . . . 4
2.3.4　後期の課題の概要 . . . . . 4
2.3.5　後期の課題 . . . . . 5
2.4　課題の割り当て . . . . . 5
2.4.1　前期の課題の割り当て . . . . . 5
2.4.2　後期の課題の割り当て . . . . . 6

**第 3 章　課題解決のプロセス** **7**

3.1　前期の課題解決のプロセス . . . . . 7
3.2　後期の課題解決のプロセス . . . . . 8

**第 4 章　班および個人による課題解決プロセスの詳細** **9**

4.1　前期の各班の課題解決のプロセスの詳細 . . . . . 9
4.1.1　Python 班 . . . . . 9
4.1.2　Unity 班 . . . . . 9
4.2　後期の各班の課題解決のプロセスの詳細 . . . . . 10
4.2.1　Asynchronous method 班 . . . . . 10
4.2.2　Deep Q-Network 班 . . . . . 11
4.2.3　Long Short Term Memory 班 . . . . . 11
4.3　個人による課題解決のプロセスの詳細 . . . . . 12
4.3.1　高橋将文 (グループリーダ, Python 班, Asynchronous method 班) . . . . . 12
4.3.2　源智也（Python 班, DQN 班） . . . . . 13
4.3.3　能登楓 (Python 班, LSTM 班) . . . . . 13
4.3.4　伊藤空（Unity 班, DQN 班） . . . . . 14
4.3.5　加藤雅崇（Unity 班, LSTM 班） . . . . . 14

4.3.6 齊藤伶奈（Unity 班, Asynchronous method 班） . . . . . 15

**第 5 章 深層強化学習の手法の説明 16**

5.1 Asynchronous method . . . . . 16
5.2 Deep Q-Network . . . . . 16
5.3 Long Short Term Memory . . . . . 17

**第 6 章 課題解決に用いた技術 18**

**第 7 章 活用した講義 19**

**第 8 章 結果 20**

8.1 年間を通しての結果 . . . . . 20
8.2 前期の成果 . . . . . 20
8.2.1 Unity 班 . . . . . 20
8.2.2 Python 班 . . . . . 21
8.3 後期の成果 . . . . . 22
8.3.1 Asynchronous method 班 . . . . . 22
8.3.2 Deep Q-Network 班 . . . . . 22
8.3.3 Long Short Term Memory 班 . . . . . 22
8.4 評価 . . . . . 23
8.4.1 中間発表での評価 . . . . . 23
8.4.2 最終発表での評価 . . . . . 24
8.5 自己評価 . . . . . 25
8.5.1 中間発表後の自己評価 . . . . . 25
8.5.2 最終発表後の自己評価 . . . . . 26
8.6 相互評価 . . . . . 27
8.6.1 前期の相互評価 . . . . . 27
8.6.2 後期の相互評価 . . . . . 28

**第 9 章 まとめ 29**

9.1 前期のプロジェクト活動のまとめ . . . . . 29
9.1.1 前期のプロジェクトの成果 . . . . . 29
9.1.2 前期プロジェクトにおける各人の役割 . . . . . 29
9.2 後期のプロジェクト活動のまとめ . . . . . 30
9.2.1 後期のプロジェクトの成果 . . . . . 30
9.2.2 後期プロジェクトにおける各人の役割 . . . . . 31
9.2.3 今後の課題 . . . . . 32

**参考文献 33**

# 第 1 章　はじめに

## 1.1　背景

深層学習とは, 近年パターン認識において優秀な成績を挙げている手法である. 注目されたきっかけとして, 画像処理の大会である ImageNet Large Scale Visual Recognition Challenge 2012(ILSVRC2012) において, 深層学習を用いたグループは前年のエラー率を 10% 近く改善した [1]. 画像処理分野だけでなく, 深層学習を用いて, 人間の認知や思考を模倣する例もある. 例えば, Google 社が開発した Google Brain では YouTube にアップロードされている動画から猫について学習し, 人の手助けなしに猫の概念を学んだ [2]. 他には, DeepMind 社が開発した AlphaGo は, 囲碁世界ランク 4 位の棋士にハンデなしの対局で勝利した [3].

（※文責: 能登楓）

## 1.2　目的

本グループでは, 人間を超える人工知能の開発を目的とする. 人工知能の学習過程, 学習結果が視覚的にわかりやすいため, 学習環境にはレースゲームを用いた. 本グループでは, 人間が操作する車に対し, 人工知能が操作するカーエージェントが一周のラップタイムを上回ることを人間を超えると定義した. また, カーエージェントとは, 周りの環境を知覚し, 車のハンドルを操作する人工知能のことである.

（※文責: 能登楓）

## 1.3　従来例

深層強化学習を用いたエージェントの例として, DeepMind 社が開発した Deep Q-Network(DQN) がある. DQN では自力でゲームを学習し, 攻略することができる. DQN はゲームのルールを教えなくても, 学習を繰り返すことによって, どのように操作すれば高得点を目指すことができるのかの戦略を学習することができる. 実際に学習したゲームは Atari2600 のゲーム 49 種類であり, そのうち, 43 種で従来研究以上の成績を出した. 中でも 29 種類のゲームでは人間のプロプレイヤー以上の性能を出した. DQN の簡易イメージを図 1.1 に示す.

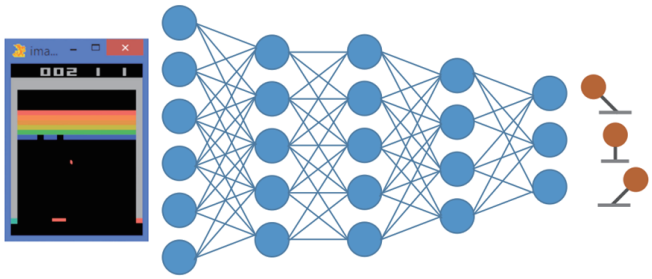

図 1.1: DQN の簡易イメージ

（※文責: 能登楓）

## 1.4 従来の問題点

DQN を用いたエージェントの問題点として大きく 2 つあげられる.

1 つ目の問題は, 報酬を得るまでに時間が掛かる問題では効果的な学習を行うことができないという点である. DQN は行動に対して報酬を得ることで学習する. しかし, DQN は学習するまで完全にランダムに行動を選択するため, 迷路のように連続した特定の行動を選択しなければ報酬を得ることができないゲームでは報酬を得ることができず, 効果的な学習を行うことは困難である.

2 つ目の問題は, 環境のうち直接観測することができない部分の情報を完全には予想することができないという点である. 例えば, レースゲームを行う際, 1 フレームのゲーム画面を見ただけでは車がどの方向にどれくらいの速度で進んでいるかを正確な数値で判断することができない. こういった直接観測することができない隠れた情報を予測するために DQN は直前の 4 フレームのみ使用する. そのため直前の 4 フレーム以前の状態やその際に行った行動に大きく依存する情報を得ることは困難である.

（※文責: 能登楓）

## 1.5 課題

人間が操作するよりも, 速く走行できるカーエージェントの開発という問題を解決するための課題は 2 つある. 1 つ目はエージェント開発に際して, 適切な深層強化学習方法の同定, 2 つ目は, 深層強化学習への入力として与える報酬, 環境の検討である.

本プロジェクトでは 2 つの課題を, グループに分け取り組んだ. また, 適切な深層強化学習手法の同定をするために, 各グループは異なる深層強化学習手法を実装した.

（※文責: 能登楓）

# 第 2 章　プロジェクト学習の概要

## 2.1　到達目標

課題解決のための総合的な到達目標を以下のように設定した.

- グループメンバとカーエージェントがそれぞれ単独でコースを 1 周し, カーエージェントがグループメンバ全員の最速ラップタイムよりも速いタイムを出す.

（※文責: 源智也）

## 2.2　問題の設定

本プロジェクトの目標は, 従来例で挙げられた DQN をカーエージェントに用いることで, 解決することができると考えた. 本プロジェクトでは, 1.4 節の二つの問題を改善することでレースゲームのカーエージェントを作成を目指す.

(1) 報酬を得るまでに時間が掛かる問題では効果的な学習を行うことができない.

(2) 環境のうち観測することができない隠れた情報を完璧に予測することができない.

（※文責: 源智也）

## 2.3　課題の設定

2. 2 節で述べた問題を解決するために設定した課題を以下に記述する.

（※文責: 源智也）

### 2.3.1　前期の課題の概要

一定の時間内に必ず報酬を得ることができるように設定し, 学習状況を状況を視覚的に観測できるようにする. レースゲームの学習環境は Unity と Python を用い, カーエージェントは Experience Replay, Fixed Target Q-Network, Long Short Term Memory(LSTM) を用いる.

（※文責: 源智也）

### 2.3.2 前期の課題

前期の課題を以下のように設定した.

- Unity での学習環境の開発
  問題 (1), (2) の解決手段として報酬, 環境を変化させた際の学習の状況を視覚的に観測できるようにすることで, より効率的に適切な報酬, 環境の設定を可能にするため.
- Experience Replay, Fixed Target Q-Network の実装
  問題 (1), (2) の解決手段として Deep Q-Network の手法を用いることでより効果的に学習を行う. 学習を効果的に行うための報酬の設定問題 (1) の解決手段として一定の時間内に必ず報酬を得ることができるように設定することで, 長時間報酬を得ることができない状態になることを防ぎ, 学習の停滞を予防するため.
- Long Short Term Memory(LSTM) を用いたネットワークの実装
  問題 (2) の解決手段として行動の判断に過去のユニットの値を用いることで, 隠れた情報をより正確に予測するため

（※文責: 源智也）

### 2.3.3 前期の反省

前期終了の段階で, 学習が成功しない原因が何であるかが正確に分からないことが問題としてあげられた. そこで, 単純な環境で単純なネットワークを用いた学習から始め, 学習に成功してから少しずつ複雑な環境で学習できるネットワークを考察していくことで, 基礎から正確に適切なネットワークを構築していくことができると考えた. そこで, 今後は OpenAI が提供している OpenAI gym を用い, 単純な環境から段階的に学習できるネットワークを構築し, 最終的にレースゲーム環境での効果的な学習を行うことが必要である.

（※文責: 高橋将文）

### 2.3.4 後期の課題の概要

一定の時間内に必ず報酬を得ることができるように設定し, 学習状況を状況を視覚的に観測できるようにする. レースゲームの学習環境は TORCS と Python を用い, カーエージェントは Asynchronous Method, Deep Q-Network, Long Short Term Memory(LSTM) を用いる. OpenAI gym を用いて単純な環境から学習させ, 最終的にレースゲーム環境での効果的な学習を行う.

（※文責: 源智也）

### 2.3.5 後期の課題

後期の課題を以下のように設定した.

- TORCS での学習環境の開発
  問題 (1), (2) の解決手段として報酬, 環境を変化させた際の学習の状況を視覚的に観測できるようにすることで, より効率的に適切な報酬, 環境の設定を可能にする.
- Asynchronous Method, Deep Q-Network, Long Short Term Memory の実装
  問題 (1), (2) の解決手段として Asynchronous Method, Deep Q-Network, Long Short Term Memory の手法を用いることでより効果的に学習を行う.
- 学習を効果的に行うための報酬の設定
  問題 (1) の解決手段として一定の時間内に必ず報酬を得ることができるように設定することで, 長時間報酬を得ることができない状態になることを防ぎ, 学習の停滞を予防する.
- OpenAI gmy を用いた段階的な学習
  OpenAI gym を用いて, 単純な環境から段階的に学習できるネットワークを構築し, 最終的にレースゲーム環境での効果的な学習を行う.

（※文責: 源智也）

## 2.4 課題の割り当て

### 2.4.1 前期の課題の割り当て

今回学習を行うネットワークを Python, レースゲームを Unity を利用して実装することから Python 班, Unity 班の 2 つの班を用意し, 個人の希望, 負荷の均一性などの基準によりメンバに割り当てた. それぞれの班のメンバへの割り当て結果は以下の通りである.

- Python 班
  高橋将文：Python-Unity 間の通信と LSTM を実装する.
  能登楓：Experience Replay を実装する.
  源智也：Fixed Target Q-Network を実装する.

- Unity 班
  伊藤空：サーキットと車を実装し, 移動に応じた報酬を計算する.
  加藤雅崇：移動に応じた報酬を計算する.
  齊藤伶奈：Python からデータを受け取り, 車に行動させる.

（※文責: 源智也）

### 2.4.2 後期の課題の割り当て

前期での Python 班での成果はそのままに, 環境設定に関してグループに対する負荷を軽減するためにオープンソースカーシミュレータの TORCS を新たに導入した. そして前期での Python 班と Unity 班を解体し, 2 人ずつ手法実装班を 3 班として再編成した. 新たな班では前期の Python 班と Unity 班が 1 人ずつになるようにし, 前期で, Unity 班に所属していた人でも, Python 班に所属していた人と協力して実装に取り組めるように編成した. 以下が新たな班割りと, 実装した手法である.

- Asynchronous method 班
  高橋将文: Asynchronous method を実装し, ハイパーパラメータを設定する.
  齊藤伶奈: 論文やインターネットで手法についての情報収集をする.

- Deep Q-Network 班
  伊藤空: ハイパーパラメータを設定する.
  源智也: Deep Q-Network を実装する.

- Long Short Term Memory 班
  能登楓: Long Short Term Memory を実装し, ハイパーパラメータを設定する.
  加藤雅崇: Long Short Term Memory を実装し, ハイパーパラメータを設定する.

（※文責: 源智也）

# 第 3 章　課題解決のプロセス

前期, 本グループでは人が操作するよりも速く走ることのできる深層強化学習を用いたカーエージェントの作成を目標に 3 つの課題を設けた. 第 1 に学習の環境となるレースゲームの作成, 第 2 にネットワークへの報酬とゲーム中の画像, 行動を送りあうための Python とレースゲーム環境の双方向通信の実装, 第 3 に Python による深層強化学習ネットワークの実装が挙げられた. また後期では, 環境を TORCS に移行し TORCS 内で取得することができるセンサ値を入力として設定したため, 第 1・第 2 の課題はなくなった. そのため, 第 3 課題である深層強化学習ネットワークの実装に重点を置き, 解決に向け開発を行った.

本章では, これらの課題に対して, 本グループがどのようなプロセスで作業を行ってきたか示す.

（※文責: 齊藤伶奈）

## 3.1　前期の課題解決のプロセス

前期では, これらの課題を効率よく解決するために Python 班 (高橋・源・能登), Unity 班（高橋・伊藤・加藤・齊藤）の 2 つの班に分けた. Python 班はネットワークの作成を行い, Unity 班はネットワークに送信するための報酬も含めたレースゲーム環境作成を行った. また Python, Unity 間の通信機能の実装は行った.

まず, 学習可能な環境, 学習可能なネットワーク作成のためのプロトタイプ作成を行った. プロトタイプ作成のために, Python 班では ILSVRC2012 で用いられた AlexNet という画像の特徴抽出を行う深層学習のネットワークを用いて簡単な深層強化学習の実装を行った. Unity 班ではレースゲームに使用するコースの作成, 既存の車アセットをベースに, 上下左右キーの入力を元に 6 方向に移動できる車を作成した. また, 両班それぞれで MessagePack と WebSocket を用いてネットワーク通信機能を実装し, AlexNet を用いたネットワークから算出された値を Unity に送信し, カーエージェットが動作する事を確認した.

プロトタイプ作成後, Python 班では適切に学習するネットワークを作るために, Experience Replay・LSTM・Fixed Target Q-Network の 3 種類の深層強化学習手法の比較, Unity 班では車のコース上の位置に基づいて車の報酬を計算する仕組みを実装, その他にネットワークが学習しやすくなるようにコースの環境を変更した. また, 図 3.1 に示す通り, 車の移動方向も設定した.

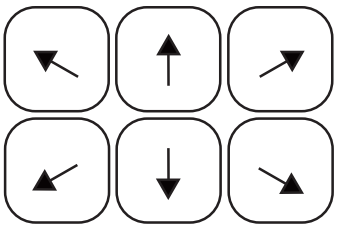

図 3.1: 車の移動方向

（※文責: 齊藤伶奈）

## 3.2 後期の課題解決のプロセス

後期ではまず, 前期の活動の反省を行った. 主な反省の内容としては 2.3.3 節に示している. ここで問題となったのが, レースゲームの画面を入力とした学習が後期中に成功するかどうかという点である. そこで, 前期の反省をまとめた後に後期の活動とスケジュールの見直しを行った. 大きな変更点として, gym-torcs という強化学習用の TORCS が公開されたため, 学習環境を Unity から TORCS というオープンソースのカーシミュレータに移行した. 理由は, 前期の活動の反省から, ネットワークの構築と学習環境の再構築を同時に行いつつ, ゲーム画面を入力としての学習が成功する可能性が低いと判断し, センサの値を用いた学習に移行するためである. この変更によって, 3 章の始めの通り, 第 1・第 2 の課題がなくなったため, 第 3 の課題を解決するためにいくつかの手法の検討を行った. また, 手法の検討を効率よく行うため, 2 人 1 組の班を 3 つ作り, 各班で別々の手法の検討を行った. 手法については, 各班がそれぞれ調査を行い, より効果的に学習できるであろう手法を検討したのち実装した.

（※文責: 加藤雅崇）

# 第 4 章　班および個人による課題解決プロセスの詳細

本章では, 第 3 章に示した課題解決のプロセス上で班および個人がどのようにアプローチを行ってきたのかを以下にまとめた. なお, 各手法の詳細については第 5 章に記している.

（※文責: 高橋将文）

## 4.1　前期の各班の課題解決のプロセスの詳細

本節では, 前期での班ごとの課題解決のプロセスをまとめた.

（※文責: 高橋将文）

### 4.1.1　Python 班

Experimence Replay, Fixed Target Q-Network, Long Short Term Memory の理解を深めそれぞれの実装を行う際には, 一人でも手法が分からなくなったらホワイトボートを用いて, 手法について話し合うことで課題解決を行った. Python と Unity 間の通信を行うために, 同様の機能を実装しているドワンゴの Life in Silico のコードを読み, 実際に実装を行った.

（※文責: 高橋将文）

### 4.1.2　Unity 班

Unity 班ではまず, 学習環境を作るため Unity を基礎から勉強し Unity 自体の扱い方の他に, スクリプトを記述する言語として C#の勉強もそれぞれ行った. その後, 既存の車アセットに対して基本的なエージェントの動作や挙動の設定を行ったが, Unity 既存のアセットではオーバルコースの作成が厳しかったため, Blender で 3D モデルとして制作し, 使用した. また, Python 班と同様に Unity 側にも Python に対して通信を行うスクリプトを記述する必要があったため, Life in Silico を参考に実装を行った.

（※文責: 高橋将文）

## 4.2 後期の各班の課題解決のプロセスの詳細

本節では, 後期での班ごとの課題解決のプロセスをまとめた.

（※文責: 高橋将文）

### 4.2.1 Asynchronous method 班

Asynchronous method 班はまず, どの手法が TORCS を学習するにあたって適切であるかを考えるために, 深層強化学習アルゴリズムや TORCS に関する論文を探して読み, お互いに知識を共有した. その結果, Asynchronous method を用いた手法が先行研究において高い性能を出しており, 他の手法を実装した論文の著者らも Asynchronous method を支持していることがわかった. そのため, Asynchronous method を利用することにした. 手法の詳細は 5.1 節に記す. また, それと並行して特別な手法を用いていない, 基本的な深層強化学習アルゴリズムを実装し, OpenAI gym で提供されている CartPole を学習させた. これは深層強化学習アルゴリズムは, 実装時に失敗してしまっていたとしてもエラーは出ないため, 早いうちに基礎の部分が正しく実装できているか確かめる必要があったからだ. その結果, CartPole を学習することができるアルゴリズムを実装することができた. このアルゴリズムのコードを他の班と共有することで, 他の班が手法の実装に早く取りかかれるようにした.

次に, Asynchronous method を正しく理解し正しく実装するために, Asynchronous method に関する論文を高橋, 齊藤の 2 人で読み, 内容について話し合うことで理解の相違をなくした.

その後, 2 人で役割分担をし, 高橋が実装, 齊藤が理論の調査を行なった. 高橋はまず, Asynchronous method を実装し, 同様に CartPole を用いて正しく実装することができているかどうかの検証を行なった. その結果, 最初に実装した基礎のアルゴリズムよりも学習速度と精度ともに大幅に性能が向上した. 次に, 齊藤が論文や書籍を読んで得た知識を利用して深層学習や強化学習におけるパラメータの調整や手法に関する工夫を行なった. 齊藤は主に深層強化学習における学習率, ユニットの初期値などのパラメータの設定方法, 最適化手法などに関する論文, 書籍を読み結果をまとめ, 高橋に報告した. このように, この齊藤が失敗の原因となっている部分に関する論文や書籍を読み, そのまとめた結果から高橋が実装するという試行錯誤を繰り返すことにより, 学習の精度や速度を向上させていった. その結果, 目標としていた性能に達することができた.

（※文責: 高橋将文）

### 4.2.2 Deep Q-Network 班

Deep Q-Network 班は, TORCS を学習させるにあたり, どのような手法があるか調べた. その調べた手法の中で, Deep Q-Network が適切だと考えた. 理由は, 1 章にある通り ATARI の複数のゲームでプロの人間以上の成果を出した実績があるからである. 加えて, スペースインベーダやブロック崩しなどの別種のゲームを全く同じネットワーク・ハイパーパラメータで学習しているため, レースゲームにも対応できると考えこの手法を選択した.

次に, 公開されている Deep Q-Network を用いたソースコードを私たちのプログラムに沿うようにプログラミングし, OpenAI gym で提供されている CartPole を学習させた. 理由は, 最初に TORCS を学習させた場合に状態の数が多くなるため学習に時間がかかり, 性能を知るのが遅くなるためである. そのため状態の数が少ない CartPole を用いることにした.

CartPole である程度成果が見込まれた後, 改めて TORCS 用にプログラミングし, TORCS 環境での学習を行った.

その後, 学習の改善を行うため Deep Q-Network, TORCS, レースゲームについて調べ, さらに学習に適するように修正した.

（※文責: 源智也）

### 4.2.3 Long Short Term Memory 班

Long Short Term Memory 班は, CartPole を学習するプログラムのプロトタイプを作成し, その後 TORCS を学習するプログラムを作成した. また, 時系列データを対象としているため, LSTM を用いてプログラムを作成した.

CartPole は, 台車 (Cart) 上の棒 (Pole) を鉛直上向きに振り上げて静止させる制御を強化学習させるゲームである. CartPole は台車の行動数が右, 左の二つしかないため, 学習の収束に時間がかからない. そのため, 短期間で何度も学習できるため, ネットワークのプロトタイプを試す環境として, 適切であると考えた. プロトタイプが成功しているかは, 強化学習で用いた報酬値の推移, 棒が立っていたステップ数を基準として, 判断した.

次にプロトタイプの入力値, 出力値を TORCS 用に変更し, TORCS を学習させた. 学習した結果, まともに走行することができなかった. 理由として, 入力値に不明な数値が多いため報酬が適切に設定できていない, 適切な出力数が定められない, などが挙げられた. そのため, TORCS について調査, プログラムの改良を二人で行った.

TORCS について調査と並行し, 報酬値の改良, ネットワークの改善を行った.

（※文責: 能登楓）

## 4.3 個人による課題解決のプロセスの詳細

本節では, プロジェクトを通して個々人がどのような課題解決のプロセスを辿ったのかを示した.

（※文責: 齊藤伶奈）

### 4.3.1 高橋将文 (グループリーダ, Python 班, Asynchronous method 班)

(1) 誰がどの様な作業をしているのか, 困っていることはないかを常に確認し問題点があればすぐに協力し解決することで, メンバ全員が作業を効率よく行えるようにし, 中間発表まで順調に作業を行うことができた.

(2) Python・Unity 間での通信を行うプログラムを実装するために MessagePack, WebSocket を用いて実装を行い, 報酬・画像・行動を送受信することに成功した.

(3) 深層学習や強化学習には様々な手法が存在ため, それぞれの手法がどのようなもので何のために行うのかを Python 班である能登・源と調べる手法を分担し, 書籍などの資料を読み, 主要なネットワークや手法, 先行研究の論文などを学習し実際に実装することができた.

(4) 基礎となる深層強化学習アルゴリズムを実装し, その学習アルゴリズムが正しく実装できているかどうかを基礎的な学習環境である OpenAI gym の CartPole を用いて検証し, 正しいと判断されたものを他の 2 つの班に配布することで, 3 つの班の手法の実装にかかる時間を短縮した.

(5) 齊藤と Asynchronous method に関する論文を読み, 話し合うことで手法に関する理解の相違を無くし, 正しく実装することができた.

(6) 齊藤が行なった学習率, ユニットの初期値などのパラメータの設定方法などに関する論文の調査の結果を受け取り, 今回の強化学習問題にどのような方法をどのように用いるかを考え, 実装を行なった. その結果, 目標となる性能に達することができた.

（※文責: 高橋将文）

### 4.3.2 源智也（Python 班, DQN 班）

(1) 目標に向けて, Python による深層強化学習の実装, Unity による環境の構築, Python と Unity 間でのデータの送受信などの課題が挙げられ, これらを満たす先行研究である Life in Silico(以下 LIS とする) のソースコードを研究することで, 効率よく機能を実装し, システムの流れを理解できるようにした.

(2) LIS のソースコードのみだと理解しにくい点, 実装できない点が存在したため, 他の深層強化学習に関連する論文やウェブサイトを調べ, 深層強化学習の実装に必要な関数の理解や実装方法を探した. そして得られた情報をネットワークを作成するメンバ同士で何度も話し合いを行った.

(3) Python 班で 1 人 1 つの深層強化学習の手法を実装することとなり, Fixed Target Q-network について学び, 実装した.

(4) 最初にレースゲームを学習させるのは難しいと判断したため, CartPole という簡単なゲームを Deep Q-Network の手法である Experience Replay と Fixed Target Q-network を用いて学習させた.

(5) DQN の手法を実装するため, 参考となるアルゴリズムを自分たちのプログラムに合うように修正した.

(6) CartPole においてある程度成果が出た後, レースゲーム用にアルゴリズムを修正した.

（※文責: 源智也）

### 4.3.3 能登楓 (Python 班, LSTM 班)

(1) 深層学習について学んだ.

(2) Chainer を用いて簡単なネットワークを作成し, 実装することができた.

(3) Experience Replay を実装して, 他手法と比較するつもりだったが, 学習がうまくいかなかった.

(4) gym-torcs を動かすために Ubuntu での学習環境を構築した.

(5) 加藤と共に RNN, LSTM について調査した.

(6) RNN, LSTM を実装し, プログラムに CartPole, TORCS を学習させた.

(7) 加藤, 他のメンバと話し合い, 報酬値の設定, ネットワークのユニット数, レイヤ数を変更した.

（※文責: 能登楓）

### 4.3.4 伊藤空（Unity 班, DQN 班）

(1) Unity を用いて学習環境を用意するために, Unity の入門本 [5] を購入し勉強した.
(2) 簡単な直線コースで学習環境を作成した.
(3) トラック状のコースを作成するための方法をインターネットで検索したり, 高橋・齊藤・加藤とレースゲームを作る上で必要な設定を Slack などを用い情報共有をした. また, ホワイトボードにアイデアを書き出して議論を行った.
(4) トラック状のコースを作成するために Blender を導入, 同ソフトを勉強し, コースを作成した.
(5) トラック状コースの報酬を計算するプログラムとゲーム画面の画像を取得する機能を実装した.
(6) ゲーム・機能のプロトタイプができ上がり次第グループリーダの高橋に報告し, 逐次フィードバックをもらうことで改善していった.
(7) gym-torcs を動かすために Ubuntu での学習環境を構築した.
(8) 源と共に学習の手法を調査し, DQN に着目して調査した.
(9) 源が書いた学習プログラムを, 構築した学習環境で動かした.
(10) 源や他のメンバと相談しながら, 適宜入力や出力, 報酬などのハイパーパラメータの調整や, DQN や強化学習のアルゴリズムの不適切な部分の修正を行った.

（※文責: 伊藤空）

### 4.3.5 加藤雅崇（Unity 班, LSTM 班）

(1) レースゲームの報酬を設定するためにゲームの設定や環境を知らなければならなかったので, ゲームの環境の実装担当者と連携して作業を行った. そして, カーエージェントが移動した距離をもとに報酬を計算させるアルゴリズムについて考えた.
(2) 移動した距離を算出するために, 現時点の車の位置座標と 4 フレーム前の位置座標を用いて計算するアルゴリズムを考案した.
(3) 能登と共に RNN, LSTM について学習, 調査を行った.
(4) 能登や他の班のメンバと共に話し合い, 報酬やハンドル操作の値を最適化できるように考え, 設定した.

（※文責: 加藤雅崇）

### 4.3.6 齊藤伶奈（Unity 班, Asynchronous method 班）

(1) Unity でレースゲームを作る上でコースの詳細な設定や, エージェントがきちんと深層強化学習できるようにゲームの要件に合わせた報酬の設定に関して班内で話し合い, それに従って作業を行った.
(2) カーエージェントが, WebSocket を介して Python から送られてきた入力毎に割り当てられた行動をとれるように車のスクリプトを改善し, 6 種類の行動を可能にした.
(3) カーエージェントの学習の妨げになると考えられた走行中のスピンを物理量を制御し, スピンしづらくした.
(4) Python と Unity で通信するために Unity で WebSocket を使えるようにモジュールを用意した.
(5) 手法を決定するために様々な論文を読み, 調べた.
(6) 手法が決定してからはハイパーパラメータをどうすれば適切に設定できるかを調べた.

（※文責: 齊藤伶奈）

# 第 5 章　深層強化学習の手法の説明

本章では, 実装した深層強化学習の手法についてそれぞれ簡単に説明を行う.

（※文責: 高橋将文）

## 5.1　Asynchronous method

オンライン強化学習アルゴリズムとディープニューラルネットワークの組み合わせは, 時系列な相関が発生してしまうことから, 基本的に学習が不安定になると考えられている. 本手法では, 多数のエージェントが並列に非同期に学習することで時系列な相関を無くし, 学習の安定化を図る. 並列化は, 複数のスレッドを用いてそれぞれのスレッドにおいて環境とネットワークの対を用意し, 学習することで行う. まず, スレッド毎に別々にもつパラメータ $\theta'$ とグローバルに共有するパラメータ $\theta$ を用意する. そして, (1)$\theta$ を $\theta'$ に同期, (2)$\theta'$ を使って $\theta$ の更新量 $d\theta$ を計算, (3)$d\theta$ で $\theta$ を更新, の 3 ステップを繰り返すことで学習を行う. また, パラメータだけでなく, 最適化手法である RMSprop の勾配の 2 乗の移動平均もグローバルに共有することで, さらに学習の安定化を図る.

（※文責: 高橋将文）

## 5.2　Deep Q-Network

Deep Q-Network は Experience Replay, Fixed Target Q-Network, Clipping の 3 つの技術を用いた手法である. Experience Replay とは, 今までに経験した各ステップにおける状態, 行動, 報酬, 次状態のセットを保存するメモリのことである. 時系列に並んでいるそれぞれのセットの並びをランダムにシャッフルしてから順に学習していくことで, 時系列な相関を無くすことができる. また, 保存することでそれらのセットを何回も繰り返し学習することができる. それによって環境との相互作用の頻度による学習速度の制限を取ることができる. Fixed Target Q-Network とは, target を固定することで収束を安定させる方法である. 強化学習における TD 誤差の target はパラメータ $\theta'$ に依存するため, $\theta'$ が不安定だと収束が安定しない. そこで, target で用いる $\theta'$ をある時点で固定し, $\theta^-$ として用いることで収束を安定させることができる. Clipping とは, 与える報酬を, 正なら 1, 負なら-1 という風に決めることで学習を安定させる方法である. その代わりに, 報酬の重み付けはできなくなってしまう.

（※文責: 高橋将文）

## 5.3 Long Short Term Memory

Long Short Term Memory はユニットが保持した値の長短期記憶を行う手法である．長短期記憶を行うために次の 3 ステップを行う．(1) ユニットに値が入力された際に, メモリに保存すべき値を求めて保存する, (2) メモリから必要なくなった記憶を削除する, (3) メモリの中から今回の判断で必要な記憶のみ読み込み今回の出力に加えて出力する．以上の処理はそれぞれ入力ゲート, 忘却ゲート, 出力ゲートと呼ばれるところで行う．これらの処理を行うことで, LSTM は短期記憶だけでなく, 長期記憶も効率的に記憶することができる．

（※文責: 高橋将文）

# 第 6 章　課題解決に用いた技術

グループ B は, 2.2 節で設定した課題を解決するために様々な技術を用いた. それらの技術を表 6.1 に示す.

表 6.1: 課題解決に用いた技術のリスト

| 技術 | 用途 | 解説 |
|---|---|---|
| ディープラーニング | 画像の特徴を抽出する | 多層のニューラルネットワークを用いた機械学習の手法 |
| 強化学習 | エージェントに最適な行動を学習させる | エージェントの状態, 報酬から最適な行動を学習させる機械学習の手法 |
| Unity | 学習環境の作成 | 3D ゲーム開発環境 |
| Blender | コースの壁の作成 | 3D モデリングツール |
| GitHub | 成果物の共有 | プログラムのバージョン管理, 共有ツール |
| Python | ネットワーク等の実装 | 機械学習用のライブラリが豊富なプログラミング言語 |
| C# | Unity のスクリプト記述 | オブジェクト指向のプログラミング言語 |
| Chainer | 深層学習アルゴリズムの実装 | 深層学習ライブラリ |

（※文責: 伊藤空）

# 第 7 章　活用した講義

本プロジェクトでは, 学習環境であるレースゲームを Unity を用いて制作した. Unity のスクリプトを記述する言語は C#を用いた. オブジェクト指向のプログラミング言語である C#を扱う上で情報処理演習 II という講義が非常に役に立った. 情報処理演習 II では, Java でのプログラミング演習を行った. このプログラミング演習で基本的なオブジェクト指向の考え方を学び, 現在 C#でのプログラミングに活用している. また, ネットワークを実装する際, 複雑な計算などは Python のパッケージを用いて省略した. しかし, 省略したまま理解を放棄するのではなく, パッケージのリファレンスを見て極力内部処理を理解しようとした. このときに応用数学 I で習った偏微分と線形代数 II で習った行列やベクトルなどの知識が非常に役に立った. また, ニューラルネットワークを勉強する際, ニューロコンピューティングという講義が非常に役に立った. 自分達で勉強していた分野の講義を履修することで, 改めてニューラルネットワークを理解することができた.

（※文責: 伊藤空）

# 第 8 章　結果

## 8.1　年間を通しての結果

3 種類の手法をそれぞれ用いたエージェントを TORCS のオーバルトラック上で走行させ, 十分に操作の練習をしたグループメンバとラップタイムの比較を行った. 手法の中でも, Asynchronous method を適用したエージェントは, 十分に学習させることで壁にぶつかることなくなめらかに走行することに成功した. その上で, 十分に操作を練習したプロジェクトグループメンバとそれぞれの最速ラップタイムを比較したところ, グループメンバ中の最速であった 1 周 1 分 37.36 秒のタイムをエージェントが 1 分 37.27 秒のタイムで超えることに成功した.

（※文責: 齊藤伶奈）

## 8.2　前期の成果

本節では前期の Unity 班, Python 班のそれぞれの成果について説明する.

（※文責: 加藤雅崇）

### 8.2.1　Unity 班

Unity 班では以下の成果が挙げられた.

- Unity の既存のアセットを用いてレースゲームに用いる車を用意した.
- 摩擦等の値の設定を確認し, より現実的な車の動きを追及した.
- Blender と Unity を用いて一種類のコース (400m トラック) のモデリングを行った. 図 8.1 に Unity の画面を, 図 8.2 に Blender で作成したコースの概形を示す.
- 学習に必要な報酬として, 車が 4 フレーム間でゴールまで道なりに進んだ距離を設定した.
- 設定した報酬とバイナリ化した画面データを Python に送れるようにした.
- Unity の画面のデータをバイナリ化し, 通信の高速化を図った.
- 設定した報酬とバイナリ化した Unity の画面データを Python に送れるようにした.
- 人間または Python からの指示やデータによって車を動かすプログラムの実装を行った.

（※文責: 加藤雅崇）

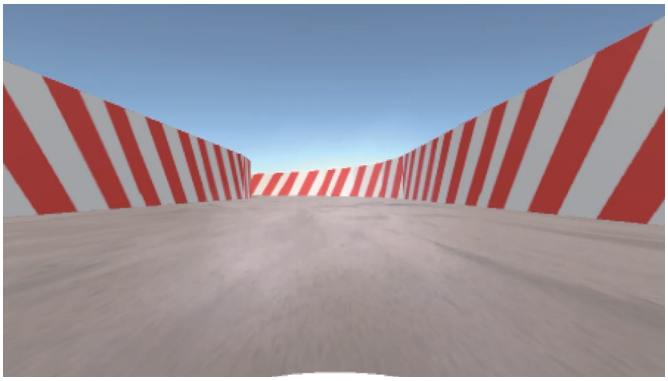

図 8.1: Unity の画面

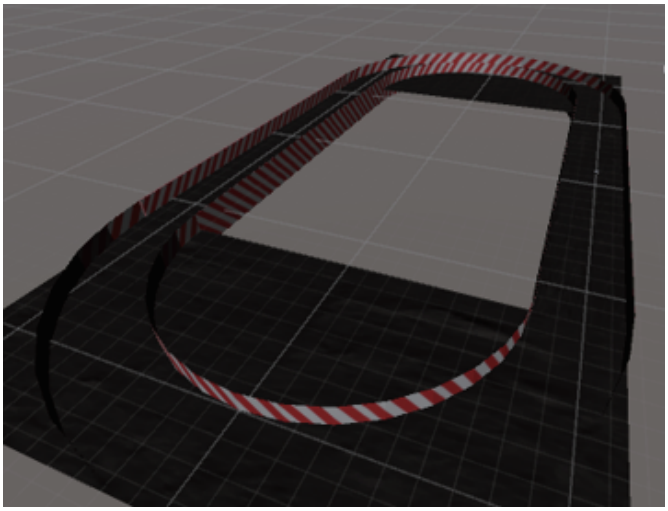

図 8.2: コース概形

### 8.2.2 Python 班

Python 班では以下の成果を挙げられた.

- 様々なネットワーク, 手法について学習し知識を獲得した.
- Unity 側から受け取ったバイナリファイルを画像ファイルに変換した .
- 画像ファイルから畳み込みニューラルネットワークを用いて特徴の抽出を行うネットワークの実装を行った.
- 特徴・報酬から LSTM を用いて行動の選択・学習を行うネットワークの実装を行った.

（※文責: 加藤雅崇）

## 8.3 後期の成果

本節では, 後期において, 3 班に分かれて行った手法実装について, それぞれの班の成果について述べる.

（※文責: 加藤雅崇）

### 8.3.1 Asynchronous method 班

- Asynchronous one step Q-Learning に関しての論文や, 非同期学習に関して様々な論文を読み技術や手法についての知識を学習した [6].
- TORCS 上のセンサの値を入力として動作するネットワークを作成した.
- Asynchronous method を実装するために, TORCS を自動で任意の数だけ同時起動できるようにした.
- CartPole 上で Asynchronous method を用いた学習を成功させた.
- OU process を用いて学習初期段階からある程度学習が円滑に進むように調整を行った.
- 学習がある程度の段階から安定に走行することを可能にした.
- 環境に適したハイパーパラメータの実装を模索した.

（※文責: 齊藤伶奈）

### 8.3.2 Deep Q-Network 班

- TORCS での車のアクセル操作を可能にした.
- GPU 用の学習プログラムを CPU での動作を可能にした.
- 学習の経過を Excel ファイルに記録する機能を実装した.
- TORCS 用の Experience Replay の実装を行った.
- TORCS 用の Fixed Target Q-Network の実装を行った.
- ネットワークの入力の改善を行った.

（※文責: 伊藤空）

### 8.3.3 Long Short Term Memory 班

- TORCS での操作を確認することによって, エージェントが行うアクセル, ハンドル操作の値を適切に設定した.
- LSTM に関する文献を調査し, TORCS 用に LSTM を実装した.
- 報酬等の値の改善を行い, 効果的に学習を行えるようにした.
- 壁にぶつかったときにリセットするといった制限を加えない状態に設定することで, オーバルコースを一周させることができた.

（※文責: 加藤雅崇）

## 8.4 評価

中間発表および最終発表に寄せられた評価とコメントを集計し, グループで考察を行った. その結果をまとめたものを以下に示す.

（※文責: 加藤雅崇）

### 8.4.1 中間発表での評価

**評価とコメントのまとめ**

中間発表で 69 人から受けた評価は, 表 8.1 に示す結果となった.

表 8.1: 中間発表での評価

| 評価 | 発表技術 | 発表内容 |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 0 | 0 |
| 3 | 2 | 1 |
| 4 | 2 | 2 |
| 5 | 5 | 3 |
| 6 | 12 | 7 |
| 7 | 16 | 16 |
| 8 | 15 | 20 |
| 9 | 7 | 9 |
| 10 | 3 | 5 |
| 無回答 | 7 | 6 |
| 平均点 | 7. 032 | 7. 476 |

また, 中間発表での発表に対して以下のようなコメントが得られた.

- 人間と機械でゲームのシステム上のハンデがあるのではないか.
- 評価方法が具体的でないため分かりにくい.
- 特定の手法に期待しすぎているのではないか.

それぞれのコメントに対して以下のような改善方法を検討した.

- ゲーム上では人間と機械が与えられてる情報の差異について検討する.
- どのように評価を行うのか数字を用いた評価方法を検討する.
- 改めて視野を広く持ち, 様々な手法について調べる.

（※文責: 加藤雅崇）

### 8.4.2 最終発表での評価

#### 評価とコメントのまとめ

最終発表で 76 人から受けた評価は, 表 8.2 に示す結果となった.

表 8.2: 最終発表での評価

| 評価 | 発表技術 | 発表内容 |
|---|---|---|
| 1 | 0 | 0 |
| 2 | 0 | 0 |
| 3 | 0 | 0 |
| 4 | 1 | 0 |
| 5 | 1 | 5 |
| 6 | 12 | 7 |
| 7 | 19 | 11 |
| 8 | 26 | 25 |
| 9 | 12 | 19 |
| 10 | 5 | 9 |
| 無回答 | 0 | 0 |
| 平均点 | 7.631 | 7.960 |
| 中間発表との差 | +0.599 | +0.484 |

発表技術に対して以下のようなコメントが得られた.

- 学習前と学習後の車の様子が比較されているデモ動画があってわかりやすかった.
- 発表に用いたスライドがわかりやすかった.
- 難しい内容をわかりやすく説明していた.
- ポスタの位置が不適切だった.
- デモ動画のモニタが角度によっては見にくかった.

発表内容について以下のようなコメントが得られた.

- 興味深い, 面白い, 楽しい内容だった.
- 技術やそれぞれの手法についてもう少し詳しく知りたかった.
- 前期からの成果が出ていた.
- アクセルをコントロールしたモデルに興味がある.
- 今後の展開が楽しみだ.

評価とコメントの分析

以上の評価とコメントから発表技術と発表内容について, 中間発表よりもわかりやすく興味を引く発表が行えたというフィードバックが得られた. またそのほかにも, このプロジェクトの将来性にも期待しているコメントも複数あり, プロジェクトの内容や今後の展望を伝えることができたというフィードバックが得られた. しかし, ポスタやモニタの位置などが見づらく不適切であったというコメントもあったので, 物の配置をあらかじめ考えて計画的に準備を進めるべきであった.

（※文責: 伊藤空）

## 8.5 自己評価

中間発表と最終発表終了後にグループ B の発表の目的と現状の把握, 今後の計画の具体性, 表現力, チームワークに対する自己評価を行った. その結果をまとめたものを以下に示す.

（※文責: 加藤雅崇）

### 8.5.1 中間発表後の自己評価

中間発表での評価を踏まえて, グループの自己評価を行った結果を表 8.3 に示す. 以下の評価は 1〜5 の 5 段階評価となっている.

表 8.3: 中間発表後の自己評価

| 項目 | 評価 | 理由 |
|---|---|---|
| 目的 | 4 | 自分たちや観客にとっても面白いという意見が多いテーマだったが, 社会的にどう役に立つかをしっかり考えていなかったため. |
| 現状の把握 | 3 | 現在効果的な学習ができない理由を明確に把握できていないため. |
| 今後の計画の具体性 | 3 | より単純な環境で行うという計画はあったが, どのような環境かという具体的な案がなかったため. |
| 表現力 | 4 | スライドや解説がわかりやすいというコメントが多かったため. ただ, 声が小さいというコメントもあったため. |
| チームワーク | 3 | 発表に向けた作業の分担が上手くできていなかったため. |

（※文責: 加藤雅崇）

### 8.5.2 最終発表後の自己評価

最終発表での評価を踏まえて, グループの自己評価を行った結果を表 8.4 に示す. 以下の評価は 1〜5 の 5 段階評価となっている.

表 8.4: 最終発表後の自己評価

| 項目 | 評価 | 理由 |
|---|---|---|
| 目的 | 4 | 観衆が興味を持ちやすいテーマであったが, 人工知能を用いて人間を超えるという課題に適切なテーマなのかといったコメントがあった. 目的に対する評価の定義が曖昧だったため自己評価を 4 点とした. |
| 現状の把握 | 5 | 3 班に分かれて実装, 文献調査を行なったが, 各班で情報の共有ができていた. また, 毎回のプロジェクト学習で各班の進捗を報告していたため自己評価を 5 点とした. |
| 今後の計画の具体性 | 5 | AI に TORCS を学習させた結果, 単純なコース (楕円形) では人間を凌駕することができた. そのため, 今後の計画として, 複雑なコースで学習する, 対戦させるなどが挙げられた. 最終発表会までには時間がなく, 断念してしまったが, 今後の計画を達成するための検討がついているため, 自己評価を 5 点とした. |
| 表現力 | 5 | 表現力は観衆から得られたコメント, 評価を基準とし, 評価している. 最終発表では, スライドに図が用いられているためわかりやすい, デモ動画を見られるので, 実際の挙動がわかるなどといったコメントがあった. しかし, 観衆にわかりやすく説明するために具体的な説明を省いてしまったため, 詳しく説明してほしいというコメントもあった. 観衆からの発表内容に関する評価は, 中間発表時よりも高い評価を得られたため, 自己評価を 5 点とした. |
| チームワーク | 4 | 発表に向けての役割分担が前期よりも上手くできため, 自己評価を 4 点とした. |

（※文責: 能登楓）

## 8.6 相互評価

中間発表と最終発表終了後, それぞれ前期と後期でのグループメンバの相互評価を行った. その結果をまとめたものを以下に示す.

（※文責: 伊藤空）

### 8.6.1 前期の相互評価

前期での相互評価のまとめを表 8.5 に示す.

表 8.5: 前期の相互評価のまとめ

| 被評価者 | 評価のまとめ |
|---|---|
| 高橋 | Python 班と Unity 班の両方に所属し, その両方で課題解決のための提案や実装, 相談に乗ることでプロジェクトメンバを引っ張っていった. また, グループリーダとして頼れる存在であった. |
| 伊藤 | Unity で作るレースゲームの多くの部分を実装し, プロジェクトに貢献した. また Unity の成果物のフィードバックから悪い点を改善していった. |
| 加藤 | Unity のレースゲームで論理的に問題の解決案を提示した. また中間発表のスライド作りに貢献した. |
| 齊藤 | グループメンバと積極的にコミュニケーションをとり, プロジェクトを良い方向に導いた. また, ポスタ作りにも貢献した. |
| 能登 | 豊富な知識を持っており, その知識を活かしてスライド作成やポスタの添削, グループメンバへの知識の提供などで貢献した. |
| 源 | 先行研究のコード解読の際に効率よく成果を出し, またわからない部分は周りと共有することで問題を解決していった. |

（※文責: 伊藤空）

### 8.6.2 後期の相互評価

後期での相互評価のまとめを表 8.6 に示す.

表 8.6: 後期の相互評価のまとめ

| 被評価者 | 評価のまとめ |
|---|---|
| 高橋 | 目的解決に向けて有益とみられる文献を調べ, 文献に記載されている手法の実装を積極的に行った. また, 他の人が躓いたときにグループリーダとして進んでその問題を解決しようとした. |
| 伊藤 | Deep Q-Network だけでなく, TORCS やレースゲームなども調べ, TORCS に適切な入力や報酬を提案・設定をした. また, 学習を行ってもらった際に行った様々な試行を分かりやすく説明したため, 次の作業の課題を円滑に考えることができた. |
| 加藤 | ネットワーク作成, 環境構築, 文献調査を積極的に行なった. また, ネットワーク作成に際して, 効率的に改善するための補助として, AI の学習を可視化するためのリアルタイムグラフの作成なども行なったため, 学習結果を直感的に理解できるようになった. |
| 齊藤 | 論文や書籍を迅速に読み効果的な手法などを提示したり, 実装の際に自分が気付かないミスを指摘したため, 問題を早期に解決していくことができた. |
| 能登 | 班分けののち, 知識の共有を行うためにわかりやすい説明を班員に行った. また積極的に情報の交換を行い, 活動を円滑に進めていた. |
| 源 | CartPole 用の DQN のプログラムやそれを TORCS 用に変えたプログラムなどを作成した. また, プログラムにエラーがでた際に原因を探ったり, 可視化してデバッグしたりした. |

（※文責: 齊藤伶奈）

# 第 9 章　まとめ

## 9.1　前期のプロジェクト活動のまとめ

本節では, 前期プロジェクト活動のまとめについて述べる.

（※文責: 高橋将文）

### 9.1.1　前期のプロジェクトの成果

本プロジェクトでは Python, Unity を用いて行動の判断, 学習を行うネットワークと実際にエージェントが行動し学習を行うレースゲーム環境を実装した. しかし, 学習が成功することはなく, 今後はネットワークとレースゲーム環境を改善していくことになった.

（※文責: 高橋将文）

### 9.1.2　前期プロジェクトにおける各人の役割

前期での本プロジェクトにおける各人の役割は以下のとおりである.

- 高橋将文 (Python 班, グループリーダ)
  Python-Unity 間での通信を行うプログラムの実装, また, Python を用いて LSTM や DQN の手法の 1 つである Clipping を用いたネットワークの実装を担当した. 結果として, 通信を行うプログラムでは, レースゲームを進めながら通信をリアルタイムで行うことに成功した. また, LSTM を用いたネットワークは, 実装し学習を始めることができたが, カーエージェントが一周するまでの効果的な学習を行うことができなかった.

- 源智也 (Python 班)
  先行研究である LIS などのソースコードの解読, また, Python を用いて DQN の手法の 1 つである Fixed Target Q-Network の実装を担当した. 結果として, LIS のコードから今回のプログラムに必要な部分を抽出することができた. また, Fixed Target Q-Network の手法の勉強を進めることができたが, 実装まで行うことはできなかった.

- 能登楓 (Python 班)
  Python を用いて DQN の手法の 1 つである Experience Replay の実装を担当した. 結果として, 手法を理解し実装に進むことはできたが, 実装で発生したバグを無くしきることはできなかった.

- 加藤雅崇 (Unity 班)
  強化学習を行う上で必要な報酬を設定するために, 4 フレーム間でカーエージェントが移動した距離を計算するアルゴリズムを担当した. 環境開発の製作を行っている伊藤と協力し,

現時点の車の位置座標と 4 フレーム前の位置座標を用いて計算するアルゴリズムの製作を行った. その結果, 一般的なオーバルコースにおける報酬の設定が可能となった.

- 齊藤伶奈 (Unity 班)
  Unity を用いてカーエジェント本体の挙動を定義するプログラムの改善, また, Python 側から行動データを受け取った際にその通りにエージェントを操作するプログラムの実装を担当した. 結果として, カーエージェント, コントローラーともに正確に実装することができた.

- 伊藤空 (Unity 班)
  深層学習の勉強会においての教師役, Unity を用いてレースゲーム・報酬を求めるアルゴリズムの実装を担当した. 結果として, 実装とともに手動での動作確認, 他のメンバへの見た目, 挙動への相談・確認を何度も行ったことで, 早い段階で実装を終了することができた.

（※文責: 高橋将文）

## 9.2 後期のプロジェクト活動のまとめ

本節では, 後期プロジェクト活動のまとめについて述べる.

（※文責: 齊藤伶奈）

### 9.2.1 後期のプロジェクトの成果

3 種類の手法をそれぞれ実装することに成功した. 手法ごとにハイパーパラメータの調整などをそれぞれ行い, TORCS のオーバルトラック上で走行することができた. 中でも Asynchronous method を用いたカーエージェントに関しては壁にぶつかることなくコースを周回することができたため, グループメンバ達と TORCS 上のあるコースでラップタイムの比較を行った. 結果, エージェントのタイムが人間のタイムを上回ることができたため 2.1 節に示した到達目標を達成することができた.

（※文責: 齊藤伶奈）

### 9.2.2 後期プロジェクトにおける各人の役割

後期での各人の役割を以下に示す.

- 高橋将文 (Asynchronous method 班, グループリーダ)
  学習に用いる PC の深層学習ライブラリやそのライブラリ上で GPU を利用するための環境構築, 基礎となる深層強化学習アルゴリズムと Asynchronous method の実装, 深層強化学習におけるハイパーパラメータや行動, 探索手法などの設定を担当した. 結果として, 目標とした性能に達するアルゴリズムを実装することができた.

- 源智也 (Deep Q-Network 班)
  CartPole に Deep Q-Network の手法である Experience Replay, Fixed Target Q-network と Clipping を実装した. そして, TORCS に Experience Replay と Fixed Target Q-network を実装した. 結果として, CartPole ではよい成果を出したが, TORCS ではコースを周回させることができなかった.

- 能登楓 (Long Short Term Memory 班)
  RNN, LSTM の理解とともに, プログラムへの実装を行なった. TORCS を学習させるために, ネットワークの改善, 報酬値の改良を行った. 結果として人間が操作するよりも速く走行させることはできなかった.

- 加藤雅崇 (Long Short Term Memory 班)
  同じ班の能登とともに実装すべき手法の検討を行い, LSTM の実装を行った. LSTM を用いたうえで, より効果的な学習を行うために, 他の班のメンバとも話し合い, 報酬や入力, ハンドル操作の値を変更し, 最適化できるように設定した.

- 齊藤伶奈 (Asynchronous method 班)
  主に, 論文を読み解くとともに, 高橋の実装の補佐を行った. 高橋とともに効果的な手法とハイパーパラメータの設定方法を調べ, 様々な論文を読んだ.

- 伊藤空 (Deep Q-Network 班)
  学習用の Ubuntu と TORCS 環境の準備と学習用プログラムのハイパーパラメータの調整を行った.

（※文責: 齊藤伶奈）

### 9.2.3 今後の課題

前期目標の画像を入力として学習させることはできなかったが, それでもエージェントは十分に練習した人間（グループメンバ）のラップタイムを超えることができた. つまり, 一番の目標であった人間を超越するという点に関しては達成できたといえる. システム上の今後の展望としては, 様々なコースへの対応, 未知のコースへの対応, 複数台の車がいる場合での学習を行うの 3 つの案が挙げられる. まず, 現在のプログラムはオーバルトラックのみに対応するように作られており, TORCS 上で実装されている他のコースに対しての性能は未知である. そのため TORCS 上の他の複数のどのコースでも対応できるようにプログラムを改善していく必要がある.

また, それに関連して現在のプログラムではある程度学習したあとのコースでなければうまく走ることができず, 例えば実際にドライバが車両に乗っているかのようにすぐにその環境に適応して走行することは困難である. よって本当に人間の性能を越えるためには初めて走るコースでもある程度対応して走行する必要がある. 初見のコースでも滑らかに走行するために, 多くの種類のコースを学習させることでプログラムの汎化性能を上げていく必要がある.

さらに, 現状のプログラムは複数台の車が同時に走行している状況には対応しておらず, 真の意味で “レース” ゲームエージェントであるとは言えない. しかし, 今回使った TORCS では車体に搭載されたセンサーの値で他車の位置を知ることもできるため, ハイパーパラメータの調整を除いて比較的容易に実装が可能だと考えられる.

プログラム以外では, 最終発表会でラップタイムの比較対象の実力がわかりづらいとの意見が多かった. つまりグループメンバの実力がわかりづらいということである. さらに, その曖昧模糊とした実力のグループメンバと比較したところで真に人間を超越したと言い切るのは説得力に欠けるため, ラップタイムのみの比較ではなく走行ラインの表示・比較を行ったり, 主観操作時の画面のブレやアクセル開度など様々な要素を加味した上で比較を行っていく必要がある. その他にもグループメンバのみでなく, レースゲーム操作の熟練者を被験者にしての比較実験を行っていく必要がある.

（※文責: 齊藤伶奈）

# 参考文献

[1]「All results」, http://imagenet.org/challenges/LSVRC/2012/results.html.(2016/07/15 アクセス)

[2] Le Q, Ranzato M, Monga R, Devin M, Chen K, Corrado G, Dean J, and Ng A, Building high-level features using large scale unsupervised learning, In ICML, 2012.

[3]『nikkei BP net』, 2016 年 3 月 31 日, 「囲碁 AI「アルファ碁」が世界トップ棋士に勝利の衝撃！ 進化する人工知能」
http://www.nikkeibp.co.jp/atcl/matome/15/325410/032800202/(2016/07/15 アクセス)

[4] 岡谷貴之, 深層学習, 講談社, 2015.

[5] 吉谷幹人, Unity5 3D/2D ゲーム開発実践入門 作りながら覚えるスマートフォンゲーム開発, ソシム, 2015.

[6] Volodymyr Mnih, Adri Puigdomnech Badia, Mehdi Mirza, Alex Graves, Timothy P Lillicrap, Tim Harley, David Silver, Koray Kavukcuoglu, Asynchronous Methods for Deep Reinforcement Learning, In ICML, 2016.

[7] Simon O Haykin, Neural Networks and Learning Machines, Pearson, 2008.

[8] Yann LeCun, Leon Bottou, Genevieve B Orr, Klaus Robert Mller, Efficient BackProp, Springer Berlin Heidelberg, 2002.

[9] Richard S Sutton, Andrew G Barto, 三上貞芳, 皆川雅章, 強化学習, 森北出版, 2000.

[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, Martin Riedmiller, Playing Atari with Deep Reinforcement Learning, NIPS Deep Learning Workshop 2013, 2013.

[11] Daniele Loiacono, Luigi Cardamone, Pier Luca Lanzi, Simulated Car Racing Championship: Competition Software Manual, 2013.

[12] Ćirović Velimir, Braking torque control using recurrent neural networks, In Proceedings of the Institution of Mechanical Engineers Part D Journal of Automobile Engineering 226(6), May 2012.

[13] Sepp Hochreiter, Jrgen Schmidhuber, Long short-term memory, In NEURAL COMPUTATION 1997.