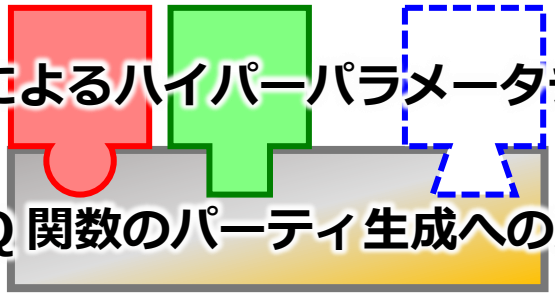
PokéAI

～ポケモンバトル AI のつくりかた～

［#4：金銀汎用行動選択モデル編］

▷あらゆるパーティの行動を操作する単一モデル

▷Optuna によるハイパーパラメータチューニング



▷強化学習 Q 関数のパーティ生成への応用

▷付録：カビゴン禁止ルール

@select766 / ヤマブキ計算所

まえがき

PokéAI（ポケエーアイ）の世界へようこそ。PokéAI は、テレビゲーム「ポケットモンスター」（以下ポケモンと表記、開発：ゲームフリーク）の対戦ゲームとしての部分、すなわちポケモンバトルに焦点を当て、その戦略を人工知能（AI）に考えさせる研究プロジェクトです。

本書は 2019 年 12 月発行の第 3 巻の続編となります。第 3 巻では、第 1・2 巻で扱った初代環境を離れ、ゲームバランスが改良され知的な戦略の有効性がより高まった環境として、ポケモン金銀（第 2 世代）のルール上で AI 開発を開始しました。ゲームのルールを再現したシミュレータと機械学習システムを接続するソフトウェア実装を行い、強化学習を実施しました。しかしながら「こうかはばつぐん」な技を選択することすら失敗することが多く、強化学習の結果に問題がありました。第 4 巻では、強化学習のモデルを大きく変更し、効率的に学習できるようにすること、そしてハイパーパラメータの調整を主に扱います。これにより、限定された環境ではありますが人が見ても妥当な結果を得ることができました。人間にとっては簡単に思いつく戦略がなかなか実現せずもどかしくなることも多いですが、AI の戦略が少しずつ進歩する様子をお楽しみいただければ幸いです。

本書の想定読者は以下のような方です。

- ポケモンを遊んだことがあって、人工知能ならどんな戦略をとるのか興味がある方
- 人工知能技術をゲームに応用する手法に興味がある方

本書は人工知能技術（特に機械学習）の知識を前提としませんが、紙面の都合上用いる技術やそれが内包する問題点をすべて解説することはできません。その場合でも最低限のアイデアが伝わるような図を用いるよう心がけています。また、過去の巻がなくても理解できるようプロジェクト概要・ポケモンバトルのルール説明等を再掲しており重複となりますがご了承ください。

なお、本書では次の事柄は扱っていません。

- ゲームの最新世代（ソード・シールド）の攻略法・リバースエンジニアリング
- ソフトウェアのインストール方法（アルゴリズム中心の内容となります）

目次

第 1 章

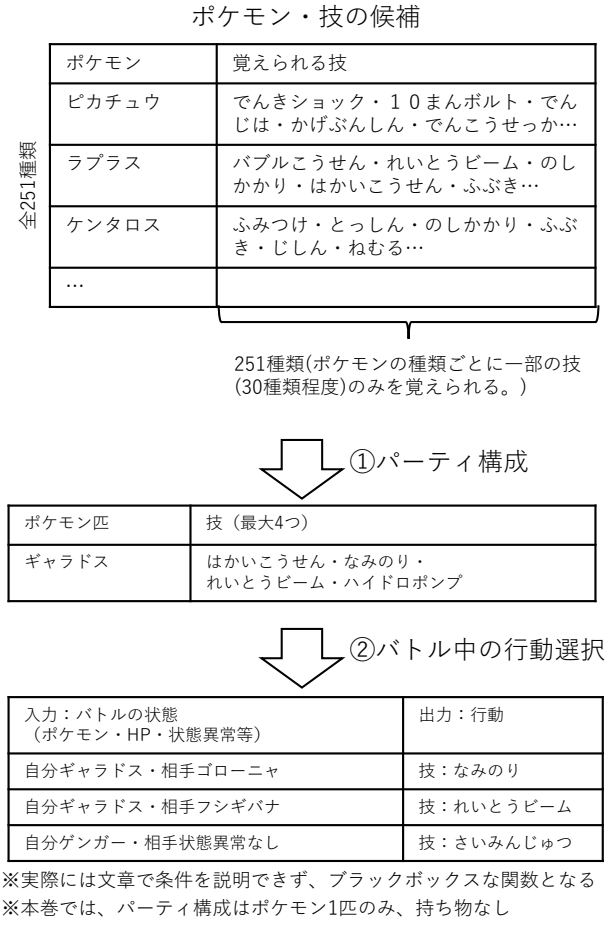
イントロダクション

PokéAI は、ポケモンバトルの戦略を人工知能（AI）に考えさせるプロジェクトです。「ふぶきが強い」というような人間の知識を極力使わずに、ポケモンバトルのルールから AI が自律的に強力な戦略を発見することを期待します。2020 年現在の人工知能技術ではこの課題の解き方は自明ではなく、他のゲームで培われた技術をもとに、ポケモンバトルに必要な技術開発を行っていく必要があります。そして、実験結果として得られた AI の戦略を鑑賞することが大きな楽しみになります。特に、ポケモンバトルは「パーティの生成」と「バトル中の行動選択」という 2 つの段階の組み合わせが必要となる点が特徴的です。

ポケモンのゲームソフト内で登場する敵トレーナーの行動を決定する機能も一種の AI です。たとえば、ジムリーダーなどの強いトレーナーは、こちらのポケモンに効果が抜群となるタイプの技を選択するようにプログラミングされています。しかし、このような AI はプログラマーが条件と行動の組み合わせを逐一設計したもの（ルールベース）であり、プログラマーが思いつかなかった戦略を発揮することはできません。本書では、条件と行動を結びつけるパラメータを人が決めるのではなく、バトルの勝敗から自動的に決定する機構を作ることで、人が思いつかなかった戦略さえも取ることが可能な AI を開発していきます。

本書では、パーティの構成と、バトル中の行動選択の両方を AI に最適化させます。全体の流れを図 1.1 に示します。筆者の知る限り、従来研究[*1]ではパーティの構成はランダムに決める、またはポケモンのタイプ補完だけ考慮するというものしか見つからないのですが、やはりこの 2 つの段階両方をうまくかみ合わせることが、ランダムにカードが配ら

[*1] Showdown AI Competition (2017): http://game.engineering.nyu.edu/showdown-ai-competition/
ポケモンバトルの AI を作ってみたよ: http://shingaryu.hatenablog.com/entry/2016/02/03/200000



▲図 1.1　ポケモンバトル AI の 2 つの課題。

れるトランプゲームや麻雀とポケモンバトルの大きな違いだと筆者は考えているため、両方を AI で検討することを目的としています。

既刊となる第 1・2 巻では、1996 年発売のポケモン赤緑（初代）のポケモンバトルのルールに準拠して AI を開発しました。機械学習技術に基づき、パーティの強さを評価する関数およびそれを用いて強いパーティを生成する手法、強化学習を用いてバトル中の行動を選択する手法を提案し、ランダムに行動するより良い結果が得られることを確認しました。しかし初代ルールはゲームバランスに難があり、活躍できるポケモン・技が狭いという問題点がありました。特に複数の技を絡めた戦法があまり有効ではなく、AI の技術を進歩させても興味深い戦法を観察できる可能性が低いという懸念がありました。第 3 巻

では、1999 年発売のポケモン金銀（第 2 世代）のルールに対応した AI の開発のための環境を整えました。オープンソースで提供されている既存の非公式ポケモンバトルシミュレータである Pokémon-Showdown と、筆者が開発する独自の AI と接続するためのインターフェースの実装を行いました。金銀ルールで新たに導入された「持ち物」について有用なものを自動的に発見するなどの結果を得た一方、バトル中の行動ではタイプ相性に基づいた技の選択も失敗する例が多いという印象でした。第 3 巻では人間同士の対戦レギュレーションに近い 3vs3、すなわちプレイヤー一人あたり 3 匹のポケモンを手持ちに入れる条件としましたが、現状の技術水準に対して複雑すぎたと考えています。第 4 巻となる本書では、より単純な 1vs1 ルールに戻り行動選択の強化学習手法を改善します。さらに、ここで学習した行動選択モデルを用いて強いパーティを生成する新たな手法を提案します。

実験に用いたソースコードはインターネット上で公開しています[*2]。

1.1 ポケモンバトルのシステム

第 3 巻とほぼ同様となりますが、ポケモンバトルの基本的なシステムについておさらいしておきましょう。ゲームのバージョンにより少し異なりますが、金銀バージョンに準拠します。また、ここで説明しきれないさまざまな例外的状況がありますが、省略しています。

ポケモンバトルは、2 人のプレイヤーがそれぞれポケモンを場に出し、技を使って相手プレイヤーのポケモンの体力（HP）を減らし（＝ダメージを与える）、先に手持ちのすべてのポケモンの HP が 0（瀕死状態）になったプレイヤーが負けというルールです。

1 人のプレイヤーは 1〜6 匹のポケモンを持ちます。このポケモンの組をパーティと呼びます。なお本巻では 1 匹のみ持つルールで行います（後述の交代という行動は存在しません）。ポケモンは 1 匹につき最大 4 種類の技を覚えることができます。ポケモンに覚えさせられる技の候補は 4 種類より多く、どの技を覚えさせておくかは戦略に依存する要素です。また、ポケモンの種類により覚えられる技は異なります。たとえば、氷タイプ[*3]のポケモンであるフリーザーはふぶきを覚えられますが、かえんほうしゃは覚えられません。また、金銀ルールではポケモンに道具を持たせることが可能です。ポケモンに持たせる道具を特に「持ち物」と呼ぶこともあります。持ち物には、水タイプの技の威力を 1.1 倍にする「しんぴのしずく」や、毒状態になった際に自動的に回復する「どくけしのみ」など約 40 種類があります。持ち物はプレイヤーがコマンドを選択して発動させるのでは

[*2] `https://github.com/select766/pokeai` 実験の都合により互換性を保たずに更新されるので、バージョンにご注意ください。

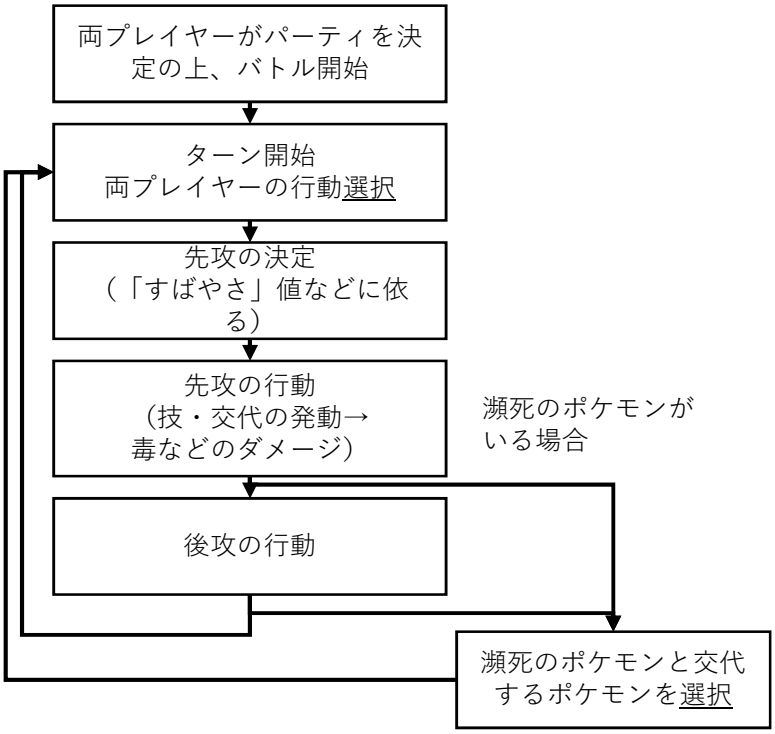
[*3] 正式なゲーム用語はひらがなですが、読みやすさやスペースの都合で漢字を用いる場合があります。

なく、発動条件が満たされると自動的に機能が発揮されます。また、相手の持ち物を奪って自分のものにする「どろぼう」という技など、直接的に持ち物に干渉するような技も用意されています。ポケモン 1 匹あたり最大 1 つの持ち物を持たせることが可能です。技と違い、ポケモンと持ち物の組み合わせ方に制約はありません。なお本巻では持ち物は扱いません。技を覚えさせたポケモンを組にしてパーティを構築するところまでは、バトルの前、すなわち対戦相手の情報を知る前に行う作業です。原作ゲームではこの部分でポケモンの捕獲・育成という過程が入りますが、本書ではゲームシステム上実現しうるすべてのパーティを使用できるものとします。また、同じ種類のポケモンであっても能力値に違いがありますが、最大値に固定します。すなわちいわゆる個体値・努力値をすべて最大値に設定します。

バトルが開始すると、両プレイヤーのパーティの先頭のポケモンが場に出ます。以後、1 プレイヤーにつき 1 匹だけ同時に場にポケモンが出ていて、技を出したり相手の技を受けたりします。

バトルはターンという単位で進行していきます。模式図を図 1.2 に示します。ターンの開始時に、両プレイヤーがそれぞれ行動を選択します。行動は、場に出ているポケモンに技を使わせるか、場に出ているポケモンを控えのポケモンと交代するかの 2 種類があります。技を使う場合、各ターンではポケモンが覚えている技の中から任意の技 1 つを選択して使うことができます（状況によっては選択肢が制限されます)。控えのポケモンとの交代は、瀕死になっていない任意のポケモンを選択して交代できます。両プレイヤーが行動を選択したら、ポケモンの「すばやさ」のパラメータ等に基づき先攻・後攻が判定され、まず先攻の行動が実行され、次に後攻の行動が実行されます。これで 1 回のターンは終了です。ターン途中にポケモンが瀕死になった場合、該当プレイヤーは控えのポケモンを選択して場に出します。あるプレイヤーのすべてのポケモンが瀕死になった時点でバトルが終了します。

ポケモンおよび技にはそれぞれタイプという属性が与えられており、たとえば「水」タイプの技が「炎」タイプのポケモンに命中すると、通常よりダメージが 2 倍になります。また、「水」タイプのポケモンが「水」タイプの技を使うと、通常よりダメージが 1.5 倍になります。さまざまな相手の弱点を突けるよう、適切なタイプの技を適切なタイプのポケモンに覚えさせておくことが戦略上非常に重要となります。また、技には相手に直接ダメージを与える「攻撃技」だけでなく、ポケモンの状態を変化させる「補助技」（変化技ともいう）があります。補助技の 1 つである「さいみんじゅつ」は、相手のポケモンを眠り状態にします。眠り状態のポケモンは、一定ターン経過して目覚めるまで技を使えなくなります。補助技は多種多様な効果のものが存在しますが、直ちに相手にダメージを与えるのではなく将来的に与えるダメージを大きくする、または自分が受けるダメージを減少させるために用います。なお、補助技にもタイプが設定されていますが、ほとんどの場合



▲図 1.2　ポケモンバトルの進行。「選択」と書かれた部分を AI が行う。

影響はありません。攻撃技と補助技の連携もまた戦略上の重要な要素となります。

1.2　本書で扱うポケモンバトルのルール設定について

第 3 巻では特殊なものを除いてポケモン金銀のすべてのポケモン・技をパーティに組み込める条件としましたが、自由度を少しでも減らして学習を容易にするためポケモン・技の数を制限した環境を用いることとしました。本節ではその手法を説明します。

バトルのルールの前提条件はこのように設定しました。

- ポケモン金銀ルール
- **パーティのポケモンは 1 匹のみ（1vs1）で、LV55 に固定、持ち物なし**
- ある程度有効なポケモン・技のみをパーティに組み込む

3vs3 は選択肢・パーティのもつ情報量が多くなり学習がうまくいかない場面が多かっ

たため、より単純な条件に設定しました。この条件で学習に成功すれば、次巻以降で緩和していきます。

従来の行動学習で扱いづらかった点として、ポケモンや技の強さに格差が大きい点がありました。たとえばトランセルはどういう戦略をとろうが勝ち目はほとんどなく、「なみのり・みずでっぽう・あわ・しっぽをふる」を覚えたカメックスではバトル中の状況にかかわらずなみのりを選択するのが最適解となるでしょう。このような状況下の行動選択データは無意味なデータとなってしまう可能性が高く、モデルの学習の妨げになると考えられます。また、トランセルの行動選択モデルの学習は純粋に時間の無駄になります。そこで、ある程度有効なポケモン・技だけを抽出することで無意味な解に陥らない環境を作成します。

まずポケモンですが、最終進化系だけを用いることにしました。ただし、禁止級伝説、技マシンが使えないポケモンを除外しています。すなわちミュウ、ミュウツー、ホウオウ、ルギア、セレビィ、メタモン、アンノーン、ソーナンス、ドーブル以外の最終進化系で、合計 129 種類となりました。他の手法として、種族値で決める方法もあるかと思います。

次に技ですが、ポケモンほど自明な基準がありません。攻撃技なら威力という基準がありますが、補助技は比較軸がありません。そこで、ランダムな技を覚えたパーティを多数生成し、第 2・3 巻で述べたパーティ評価関数の手法で技の有用度を定量化することとしました。

ポケモンは最終進化系だけをランダムに選択し、技はそれぞれのポケモンが覚えられるものからランダムに 4 つ選択します。このようなパーティを 10,000 個生成し、ランダムに行動させた勝敗から各パーティのレートを計算します。ここでは技がレートに与える影響を抽出したいので、パーティ特徴量として技（M）特徴量のみを用います。

技ごとに算出された係数の最高 10、最低 10 を表 1.1 に表示します。係数が大きいほど勝利に貢献していることを表します。

▼表 1.1　技ごとのパーティ評価関数の係数

技	係数
はかいこうせん	0.660
じしん	0.561
なみのり	0.493
１０まんボルト	0.477
れいとうビーム	0.460
のしかかり	0.418
かいりき	0.415
ハイドロポンプ	0.406
かえんほうしゃ	0.393
ちきゅうなげ	0.392
...	...
あまごい	-0.411
やつあたり	-0.411
いびき	-0.423
ねごと	-0.449
こらえる	-0.456
しんぴのまもり	-0.479
メロメロ	-0.484
だいばくはつ	-0.490
じばく	-0.760
いとをはく	-0.862

はかいこうせん、じしんなどの強そうな技が上位に来ています。下位では、1vs1 バトルなので使うと即負けになるじばくのほか、使用条件を満たさないと無意味なメロメロ、ねごとなどが出ています。これらの技の有効な運用もいつかは実現したいですが、「こうかはばつぐん」の技を選ぶことすら苦労している現状なので後回しです。

ポケモンごとに技構成の自由度をもたせつつ無意味な技を排除するため、各ポケモンの覚えられる技のうち係数上位 8 個を選択し、全（最終進化系）ポケモンにわたって和集合をとりました。その結果、リスト 1.1 の 52 個の技が選択されました。

▼リスト 1.1　パーティ評価関数をもとに選択された技リスト

```
１０まんボルト，いあいぎり，いわくだき，おんがえし，かいりき，かえんほうしゃ，かげぶんしん，かみなり，かみなりパンチ，ギガドレイン，げんしのちから，ゴッドバード，ころがる，サイケこうせん，サイコキネシス，じしん，ずつき，すてみタックル，すなあらし，スピードスター，ソーラービーム，そらをとぶ，だいもんじ，たきのぼり，ちきゅうなげ，つのでつく，つのドリル，つばさでうつ，でんじほう，どくどく，どくのこな，とっしん，ドリルくちばし，どろかけ，ナイトヘッド，なみのり，のしかかり，ハイドロポンプ，はかいこうせん，はがねのつば
```

さ，ばくれつパンチ，はっぱカッター，はなびらのまい，バブルこうせん，ふぶき，ふみつけ，ヘドロばくだん，ほのおのパンチ，やどりぎのタネ，れいとうパンチ，れいとうビーム，ロケットずつき

以上、ルール上存在するポケモン・技のうち、ランダムに選択しても大きく格差が出ないよう有効なポケモン 129 種類・技 52 種類からなるサブセットを抽出しました。この範囲で汎用行動選択モデルの学習を進めていきます。

まったく別の手段として、人間同士の公式大会（ニンテンドウカップ）で使われたポケモン・技に絞るというやり方もあるかと思います。人間の思考に影響されてしまうとはいえ、ランダムに技を使った時の効果で測定するよりは多様な補助技を含められることが期待できます。

1.3 汎用行動選択モデル

本節では、バトル中の行動選択を行う AI が持つべき機能と、それを実現するため本巻で新たに提案する構造を説明します。

バトル中の行動選択を行う AI は、バトルの各ターンにおいてポケモンの残り HP などの「状態」を受け取り、「行動」としてそのターンで使う技（本巻のルールでは、交代なし）を出力します。複数ターンにわたるバトルで最終的に勝利することが目的となります。強化学習の分野では、状態を受け取り行動を返すという意思決定を行う機構を「エージェント」と呼びます。ゲームソフト側は「環境」と呼ばれ、エージェントから行動を受け取って技などの発動処理を行い、1 ターン進んだ後の状態をエージェントに返します。エージェントと環境が相互作用することによりバトルが進行します。さらに、環境は各ターンで状態とともに「報酬」という数値を返します。バトルにおいては決着がついたときにエージェントが勝ちなら +1、負けなら-1 という値になります。決着がついていないターンでは 0 です。報酬はバトルの進行そのものには関係なく、エージェントのパラメータを調整するために使う値です。強化学習とは、エージェントが試行錯誤しながら、エージェントがバトルを通じて得られる報酬を最大化するようにパラメータを調整する技術です。最も一般的な強化学習では環境に対してエージェントは 1 つですが、ポケモンバトルでは 2 人のプレイヤーが対戦するので、エージェントは 2 つになります。2 つのエージェントはそれぞれ自分の側から見た状態や報酬を受け取ります。

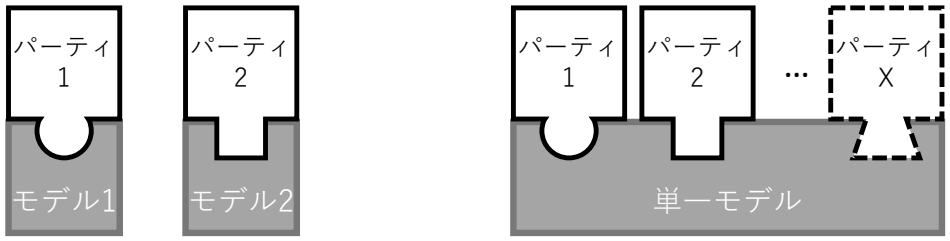
ここまでで説明したエージェントは抽象的なものでしたが、コンピュータのプログラムとして実現する必要があるので、具体的な計算手順を定める必要があります。本書ではこれを「モデル」と呼びます。もっとも単純なモデルは常に 1 番目の技を選ぶというもので、これでも状態を受け取って（そしてそれを無視して）行動を返すというエージェント

の役割を果たせます。もちろんこれでは面白くないので、状態に応じて行動が変化するようなモデルを考える必要があります。たとえば、相手が水タイプなら電気タイプの攻撃技を選択する、というルールを羅列するルールベースの手法があります。本書の目的は、人間が考えたルールに依存せず強いエージェントを自動的に作ることなので、報酬からのフィードバックで行動の判断基準を自動的に調整できるようなパラメータを持つ必要があります。そのようなモデルの 1 つとして、行列演算を用いることができます。状態を何らかのベクトル s で表現し、パラメータとして行列 W を用意します。そして、行列積 Ws を計算します。行動 a の選択肢が 4 つあるとすると、$a \in \{1, 2, 3, 4\}$ に対してベクトル Ws の a 番目の要素がその行動の優先度を表すことにします。ここで優先度とは、実数値で大きいほどその技を選択すべきであることを表します[*4]。同じ s に対しても W によって異なる結果となるので、W の具体的な値を強化学習によって調整します。なお、本書で実際に用いるモデルは、行列積を 1 回だけではなく、複数回行うことで複雑な入出力関係を表現できる深層ニューラルネットワーク（DNN）となります。

今までポケモンバトル中の行動選択を行うモデルは、パーティごとに別個のパラメータを学習していました。しかし今後の発展を考えるとこの方式は難点があり、あらゆるパーティの行動選択を行える単一のモデルを学習することを考えます。

従来の行動選択モデルは、パーティごとに別個のパラメータを学習します。モデルの入力はバトル中の状態で、相手のポケモンの種族、自分と相手それぞれの HP の残り比率、ランク補正状態などです。モデルの出力は技 1〜4 それぞれの優先度を表す値となります。本巻では、ポケモン 1 匹だけをパーティに組み込む 1vs1 バトルで考えます。モデルの入力として、自分のポケモンの種族やどの技を覚えているかという情報はありません。技 1〜4 が何であるか知らずとも、水タイプの相手に最大ダメージを与えられるのは技 1 で、炎タイプの相手に最大ダメージを与えられるのは技 3 であるというような情報を勝敗から学習していけば十分であるためです。この方式の問題点は、パーティ構成が変われば一から学習しなければならないという点です。プロジェクトの大きな目標として、パーティ構成とバトル中の行動選択の両方を最適化したいと考えています。さまざまなパーティを試行錯誤して作る際、過去に学習した行動選択モデルの情報が使えれば、学習コストを下げられる可能性があります。自分のパーティ構成がなんであろうと、水タイプの相手には１０まんボルトやメガドレインが有効である、というような知識は有用です。そのような知識をどのパーティでも共有し、新しく生成されたパーティでも、１０まんボルトとれいとうビームを覚えているなら相手が水タイプのときは１０まんボルトを選ぶということを行動選択モデルの学習なしに実現できれば、さまざまなパーティの強さを（ランダムに行動

[*4] でんこうせっかが素早さに関係なく先制できるという意味での優先度ではないことにご注意ください。より誤解の少ない日本語があるといいのですが…



▲図 1.3　パーティ固有行動選択モデルと汎用行動選択モデルの比較。

させるより正確に）評価することが容易になります。パーティの技 1〜4 の効果が固定という前提が取り払われれば、別のメリットとして、「へんしん」等で自分の技が変化した場合にも対応することができるようになります。ただし、タイプ相性すら難儀している現状では難しすぎる課題のため、相手に応じて技の挙動が変化するような特殊な技には取り組みません。

これを実現するために、自分のパーティにかかわらず 1 つのモデルパラメータを共有する「汎用行動選択モデル」を学習したいと思います。これに対して従来の行動選択モデルは「パーティ固有行動選択モデル」と呼ぶことにします。汎用行動選択モデルでは、入力として自分のパーティのポケモンの種族と覚えている技も与えて、ターンごとにどの技を使うかを出力します。

パーティ固有行動選択モデルと汎用行動選択モデルの違いを表す模式図を図 1.3 に示します。パーティ固有行動選択モデルでは、パーティが決まった上でそれを運用するためのパラメータを一から学習します。異なるパーティに対応するモデルは情報を何も共有していません。汎用行動選択モデルは、自由度の高い（学習対象となるパラメータ数の多い）モデル 1 つを学習し、多数のパーティに対応します。対応するパーティは学習時に出現した物だけでなく、未知のパーティも含みます。たとえば学習時のパーティとして「ハピナス＋どくどく＋かげぶんしん」および「ブラッキー＋どくどく＋かげぶんしん」がいて、いずれもどくどくを使ったあとにかげぶんしんを使うという戦略が有効であれば、学習時には出現しなかった「エアームド＋どくどく＋かげぶんしん」でも同様の戦略が有効かもしれない、と推測できます。汎用行動選択モデルでは、類似したパーティでは類似した戦略で行動させることが可能となります。もちろん、学習データに一度も出現していないポケモンや技があればうまくいきません。システム上、技はただの番号で管理されます。た

とえばわざマシン 01 が「ばくれつパンチ」で 02 が「ずつき」のときに 03「のろい」の効果を予想する、ということはできません。あくまで見たことのあるポケモンや技について、新しい組み合わせでも既存の知識が活用できるよう、機械学習により汎化能力をもたらすことが目標です。また、耐久力が低いポケモンであるマルマインではどくどく＋かげぶんしんという戦略はハピナスほど効果がない、ということを考慮するには同じ技であっても耐久力の高低、相手とのタイプ相性などで有効性がどう変化するかが学習される必要があります。どれだけ多様な組み合わせに対応できるかは学習データの分量やモデル設計の良し悪しで決まります。

本章では、ポケモンバトル AI の概要を紹介したのち、バトル中の行動選択エージェントに用いるモデルとして過去の巻で用いていたパーティ固有行動選択モデルに代わり、汎用行動選択モデルを提案しました。従来は自分のパーティごとに異なるエージェントを学習する仕組みであり、自分のパーティごとに別々の環境とエージェントの組があるということになります。汎用行動選択モデルでは自分のパーティにかかわらず環境もエージェントもただ 1 つだけが存在し、バトル開始時にエージェントが担当すべきパーティを環境がランダムに選択し、状態中に自分のパーティ構成を含めるという仕組みとなります。第 2 章では、いきなり汎用行動選択モデルを強化学習させるのは難しいと判断し、問題に適したモデルの複雑さを探るための教師あり学習を行います。第 3 章では、教師あり学習で定めたものと同一の複雑さのモデルを強化学習させ、より強いエージェントの実現を目指します。第 4 章では、学習したモデルを応用して強いパーティを生成する手法を提案します。第 5 章では、強いパーティの生成とそのバトル中の行動の強化学習を交互に行い、ポケモンバトル全体における戦略を最適化します。第 6 章では、結論と今後の展望を述べます。

第 2 章

汎用行動選択モデルの擬似教師あり学習

パーティ固有行動選択モデルではあくまで自分が覚えている技 4 つのどれを選択するかだけを考慮すればよかった一方、汎用行動選択モデルでは 100 種類以上存在する技・ポケモンすべての運用を知っている必要があるため、学習すべきパラメータ数が増加します。また、どのようなモデル構造（深層ニューラルネットワークの構造）にすればいいかも明らかではありません。パーティ固有行動選択モデルでも勝敗から強化学習した結果のモデルが「こうかはいまひとつ」な技を選択してしまうなど難ありの状況のため、パラメータ数が増加したモデルの強化学習は難しいと予想しました。この課題を解決するアイデアとして、強化学習よりも容易である教師あり学習で汎用行動選択モデルを学習させ、どのようなモデル構造・パラメータ数の規模ならうまく学習できるのかを検討することにしました。強化学習では複数ターンかかるバトルの終わりに判明する勝敗だけを手掛かりにパラメータを調整する一方で、教師あり学習は各ターンにおいて正解となる行動を与えてパラメータを調整する技術であり、最適化がより単純になります。ただし、教師あり学習には教師データ、すなわちバトルの状況と適切な行動（技）の組が必要です。しかしそのようなデータはオンライン対戦サービスを運営していない限り自動的に収集できるものではないため自作する必要があります。ひとつの方法は人力で作成することですが、数千件必要であろうサンプルを作るのが大変そうだし条件が変わるたびに作り直しになるので不採用です。そこで今回は、パーティ固有行動選択モデルを多数のパーティに対して学習させ、モデル間で対戦させてバトルの状態とモデルが選んだ行動の組を収集し、疑似的な教師データとして用いることにしました。ゲーム AI としてこのような手段を用いた事例があるかは分かりませんが、10 万種類の画像分類を行うモデルを学習するため、自動車ジャンルだけ、鳥ジャンルだけといった特化モデルを学習し、その知識をひとつの汎用モデルに

吸収させる研究があります[*1]。

2.1 擬似教師データの作成

この節では、モデルの構造を教師あり学習で検討するための疑似教師データの作成を考えます。

汎用行動選択モデルとして深層ニューラルネットワーク（DNN）を用いますが、パーティの情報を取り入れるための構造や、パラメータ数の自由度が非常に大きいため適切な規模のものを選ぶ必要があります。モデルを強化学習させると強化学習自体にもさまざまなハイパーパラメータが必要（報酬割引率や replay buffer のサイズ等）で、探索が大変です。ハイパーパラメータとは、学習を通じて調整されるパラメータとは別に、学習機構の制御を行ったりモデルのパラメータ数を決めたりするパラメータです。そのため強化学習の適切なハイパーパラメータがわかっているパーティ固有行動選択モデルを用いて局面ごとの行動の正解を生成します。パーティ固有行動選択モデルは 1 パーティに対し 1 つのモデルが対応します。さまざまなパーティに対しそれぞれパーティ固有行動選択モデルを強化学習させ、その行動データを集積し、それを 1 つの汎用行動選択モデルに教師あり学習させるという流れを提案します。パーティ固有行動選択モデルにより選択された行動を疑似教師データと呼ぶことにします。「疑似」とついているのは、パーティ固有行動選択モデルの強化学習が完ぺきではなく、必ずしも最適な行動を選択できるわけではないということを指しています。

疑似教師データの作成は次のように実装しました。パーティをランダムに 1,000 個生成させ、それぞれに対してパーティ固有行動選択モデルを学習させます。パーティの条件は「1.2 本書で扱うポケモンバトルのルール設定について」で検討した最終進化系ポケモンだけを含むものです。強化学習中の対戦相手は自分以外のパーティがランダムに行動するエージェントです。こうして学習させたエージェント同士を対戦させ、対戦ログを保存する仕組みを用意しました。対戦ログには、各ターンにおける状況（相手のポケモンの種族、残り HP、状態異常など）・自分のパーティの情報（ポケモンの種族、覚えている技など）・エージェントが選択した行動が含まれます。

実際のログを整形していくつか表示します。

▼リスト 2.1　疑似教師データの凡例

[*1] Jiyang Gao et al., Knowledge Concentration: Learning 100K Object Classifiers in a Single CNN. arXiv 1711.07607.

```
自分のポケモン 現在HP/最大HP 状態異常等
自分のポケモンの技構成
相手のポケモン 現在HP/最大HP 状態異常等
=> 選択した行動
```

▼リスト 2.2　疑似教師データの例

```
自分 マタドガス 187/187
ころがる １０まんボルト でんじほう だいもんじ
相手 オムスター 193/193
=> １０まんボルト

自分 ラフレシア 198/198
はかいこうせん やどりぎのタネ ギガドレイン おんがえし
相手 ウソッキー 193/193
=> ギガドレイン

自分 ラプラス 259/259
いわくだき １０まんボルト サイコキネシス ハイドロポンプ
相手 オクタン 198/198
=> サイコキネシス

自分 ブーバー 187/187
すてみタックル サイコキネシス いわくだき ずつき
相手 ジュゴン 215/215
=> いわくだき
```

必ずしも最適とはいえないですが、ある程度「こうかはばつぐん」な行動を選べています。

次の例は 1 つのバトル中の各ターンの行動を示したものです。

▼リスト 2.3　1 つのバトルから抽出した疑似教師データの例

```
自分 プテラ 204/204
かげぶんしん つばさでうつ どくどく だいもんじ
相手 ポリゴン２ 209/209
=> どくどく

自分 プテラ 157/204
かげぶんしん つばさでうつ どくどく だいもんじ
相手 ポリゴン２ 185/209 tox
=> だいもんじ

自分 プテラ 108/204
かげぶんしん つばさでうつ どくどく だいもんじ
相手 ポリゴン２ 106/209 tox
=> かげぶんしん

自分 プテラ 63/204   evasion+1
かげぶんしん つばさでうつ どくどく だいもんじ
相手 ポリゴン２ 56/209 tox
```

```
=> かげぶんしん

自分 プテラ 63/204   evasion+2
かげぶんしん つばさでうつ どくどく だいもんじ
相手 ポリゴン2 4/209 tox
=> だいもんじ
```

tox は猛毒状態、evasion+1 は回避率が 1 段階上がった状態を表します。このエージェントはどくどくのあとかげぶんしんで逃げ回るという戦略をとっていることが読み取れます。このようにパーティ構成と相手のポケモンだけで技が決まるわけではなく、相手の状態異常といったバトル中の状態に依存して決めることが学習されている場合もあります。

2.2 汎用行動選択モデルの設計

前節で作成した教師データを用いて、汎用行動選択モデルの学習を試みます。自分のパーティの情報を含むバトルの状態を入力とし、適切な行動（技）を選択するモデルを学習することが目標です。

モデルとして DNN を用います。DNN は畳み込み、リカレントなど構造の自由度が極めて高いですが、今回はもっとも単純に f(バトルの状態, 選択肢 i の情報) => 選択肢 i の優先度 という入出力の fully-connected feedforward network にすることとしました。

バトルの状態のベクトルでの表現方法を表 2.1 に示します。パーティ固有行動選択モデルで用いたもの[*2]に加え、自分のパーティ構成を与える必要があります。自分のパーティ構成として、ポケモンの種族を与えることとしました。技については選択肢ベクトルに含まれるので、重複を避けてこちらには含めないこととしました。厳密なことをいえば、炎タイプの攻撃技を持っている場合ににほんばれの優先度を上げるというような機能を実現するためには、選択肢の技以外に何を覚えているかを情報として与える必要がありますので、将来的には入力特徴量の情報を増やして表現力を上げることが望ましいです。

[*2] 3vs3 バトルと同等のものを用いていますが、1vs1 のため選択肢は技 4 つだけで、生存ポケモン数などは意味がありません。

▼表 2.1　バトルの状態を表現する特徴量。自分/相手は、どちら側のパーティの情報を入力として与えるか。両方の場合は次元数が倍となる。

特徴	次元数	自分/相手	説明
有効な行動	4	自分	現在どの行動がとれるのかを表す 有効な行動に該当する次元に 1 を設定
生存ポケモン数	1	両方	瀕死でないポケモン数/全ポケモン数
ポケモンタイプ	17	相手	場に出ているポケモンのポケモンのタイプ （ノーマル・水・…）に該当する次元に 1 を設定
HP 残存率	1	両方	場に出ているポケモンの現在 HP/最大 HP
状態異常	6	両方	場に出ているポケモンの状態異常（どく・もうどく・ まひ・やけど・ねむり・こおりのうち 該当次元に 1 を設定)
ランク補正	6	両方	場に出ているポケモンのランク補正（こうげき・ ぼうぎょ・とくこう・とくぼう・すばやさ・命中・ 回避それぞれ、ランク/12+0.5 を設定）
天候	3	-	場の天候（はれ・あめ・すなあらし）に 該当する次元に 1 を設定
ポケモン種族	251	自分	場に出ている自分のポケモンの 種族に該当する次元に 1 を設定

モデルに与える選択肢を表現するベクトルとして今回はもっとも単純に、技を表す 251 次元（技はポケモン数と同じ 251 種類存在）のベクトルを用いました。技の番号に対応する次元に 1 を設定します。実際にパーティに含める技は前述のように 52 種類に絞っているので、出現しない技に対応する次元の値は常に 0 です。バトルの状態ベクトルと選択肢ベクトルを連結したもの、合計 558 次元を DNN への入力とします。

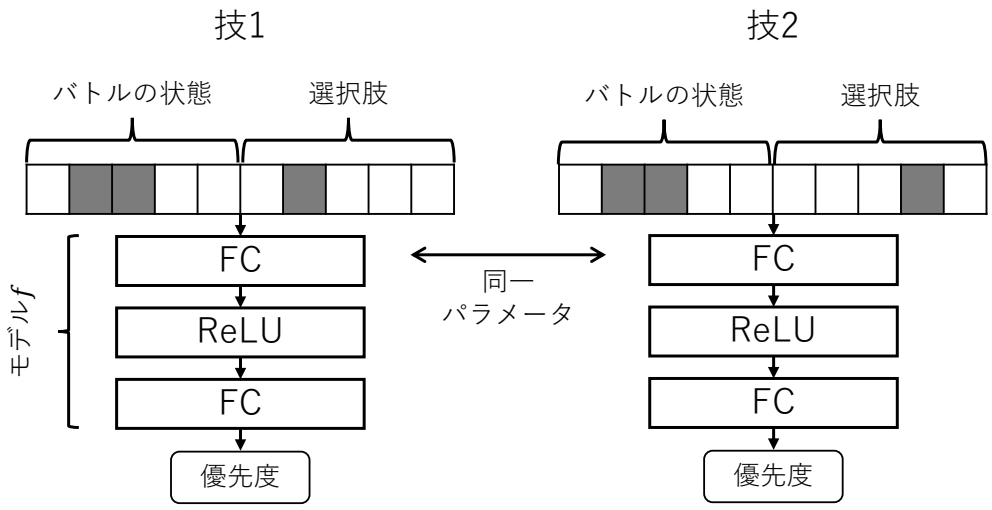
DNN を用いたモデル構成は図 2.1 のようになり、深層学習ライブラリ PyTorch を用いた定義コードをリスト 2.4 に示します。

▼リスト 2.4　DNN の PyTorch 実装例（隠れ層 1 層の場合）

```
import torch.nn as nn
import torch.nn.functional as F

class MLPModel(nn.Module):
    def __init__(self, input_dim):
        super().__init__()
        self.fc1 = nn.Linear(input_dim, 64)
        self.fc2 = nn.Linear(64, 1)

    def forward(self, x):
        h = x  # batch, feature_dim
        h = F.relu(self.fc1(h))
        h = self.fc2(h)
        return h
```



▲図 2.1　モデル・入出力の構成。選択肢（技）ごとに異なる選択肢ベクトルをモデルの入力とし、選択肢の優先度を計算する。

教師あり学習をするにあたり、一般的な分類問題の定式化に落とし込みます。$softmax(f(\text{バトルの状態}, \text{選択肢 1 のベクトル}), f(\text{バトルの状態}, \text{選択肢 2 のベクトル}), f(\text{バトルの状態}, \text{選択肢 3 のベクトル}), f(\text{バトルの状態}, \text{選択肢 4 のベクトル}))$ を各選択肢の選択確率として、正解データとの cross entropy loss を最小化するように学習します。

先ほど定義したモデルに選択肢ベクトルを変えて 4 回呼び出し、結果を連結して損失を計算することも可能ですが、実装上あまり効率がよくありません。そこで、モデルの呼出しを 1 回ですべての計算を終えるテクニックがあります。それは、全選択肢に対応するベクトルを積み重ねた行列（558×4）を入力とし、全結合層をカーネルサイズ 1 の 1D Convolution（畳み込み）に置き換えることです。形式的には畳み込みですが、畳み込みとしての性質はなく一気に全選択肢分のベクトルに対して同一の計算が適用でき、計算結果は等価です。この技法は画像処理で用いられる「1x1 convolution」に近いです。

▼リスト 2.5　1D Convolution により一気に全選択肢の計算をするコード

```
import torch.nn as nn
import torch.nn.functional as F

class MLPModel(nn.Module):
    def __init__(self, input_dim, n_layers=2, n_channels=64, bn=False):
```

```
        super().__init__()
        layers = []
        bn_layers = []
        cur_hidden_ch = input_dim
        for i in range(n_layers):
            layers.append(nn.Conv1d(cur_hidden_ch, n_channels, 1))  # in,out,ksize
            cur_hidden_ch = n_channels
            if bn:
                bn_layers.append(nn.BatchNorm1d(n_channels))
        self.layers = nn.ModuleList(layers)
        self.bn_layers = nn.ModuleList(bn_layers)
        self.output = nn.Conv1d(cur_hidden_ch, 1, 1)
        self.bn = bn

    def forward(self, x):
        h = x  # batch, feature_dim, 4
        for i in range(len(self.layers)):
            h = self.layers[i](h)
            if self.bn:
                h = self.bn_layers[i](h)
            h = F.relu(h)
        h = self.output(h)
        h = h.view(h.shape[0], -1)  # batch, 4
        return h
```

2.3 実験

このように定義したモデルを学習させます。パーティ 1,000 個（それぞれに対するパーティ固有行動選択モデルを学習済み）を 1 パーティ当たり 100 回他のパーティと対戦させ、10 万バトル分の行動を得ました。同じバトルについて 2 つのパーティから見た状態を別のデータとして扱っています。900 パーティ分のデータを学習データ、残り 100 パーティ分を評価（validation）データとして使用します。1 回のバトルで複数のターンがあるので、学習データは 34 万サンプルとなりました。

上記のモデルのハイパーパラメータを変更して正解率を測定しました。学習は 10 エポック、Optimizer は Adam、learning rate=0.01、バッチサイズ 256 としました。層数は出力層以外の Convolution の数です。チャンネル数は隠れ層の出力チャンネル数です。バッチ正規化は、各 Convolution の後に学習を安定化させる Batch Normalization レイヤーを付加するか否かです。

▼表 2.2　モデルのハイパーパラメータと正解率

層数	チャンネル数	バッチ正規化	Training 正解率 [%]	Validation 正解率 [%]
1	16	False	55.2	52.5
1	16	True	67.9	55.6
1	64	False	58.2	54.6
1	64	True	72.5	55.2
1	256	False	63.8	53.8
1	256	True	75.1	53.1
2	16	False	65.6	54.2
2	16	True	68.6	56.4
2	64	False	70.5	55.9
2	64	True	73.2	54.6
2	256	False	70.1	55.4
2	256	True	**76.2**	53.2
3	16	False	66.2	**58.5**
3	16	True	67.9	56.0
3	64	False	68.2	57.8
3	64	True	73.6	54.4
3	256	False	68.6	54.8
3	256	True	75.9	53.6

Validation 正解率が最大となるのは 3 層、チャンネル数 16、バッチ正規化なしという結果になりました。ハイパーパラメータ間に極端な差はないですが、層の数は多いほうがよい一方で、チャンネル数が多いと Training 正解率は高くなる一方で Validation 正解率は下がってしまい過学習していることがわかります。バッチ正規化についても、過学習を誘発しているようです。

2.3.1　学習したモデルの定性評価

学習した汎用行動選択モデルの挙動を定性的に確認してみます。

定量的にもっとも精度が高かったモデル（3 層、チャンネル数 16、バッチ正規化なし）に対し Validation データを入力し、モデルの出力（選択した技）および正解データを表示します。正解データはパーティ固有行動選択モデルが出力したものです。

▼リスト 2.6　凡例

```
自分のポケモン
自分のポケモンの技構成
相手のポケモン
=> 各行動に対する確率
モデル出力=最大確率の技 正解=正解データ
```

▼リスト 2.7　モデル出力の例 1

```
自分 ドククラゲ 204/204
ハイドロポンプ おんがえし とっしん ヘドロばくだん
相手 スターミー 182/182
=> ハイドロポンプ=2.4% おんがえし=7.3% とっしん=0.2% ヘドロばくだん=90.1%
モデル出力=ヘドロばくだん 正解=ヘドロばくだん

自分 ランターン 52/253
なみのり どくどく おんがえし でんじほう
相手 ドンファン 80/215
=> なみのり=89.0% どくどく=3.1% おんがえし=5.4% でんじほう=2.5%
モデル出力=なみのり 正解=なみのり
```

妥当な出力をしています。

▼リスト 2.8　モデル出力の例 2

```
自分 ゴローニャ 138/204 psn
すてみタックル ちきゅうなげ じしん ずつき
相手 パラセクト 25/182
=> すてみタックル=10.7% ちきゅうなげ=18.0% じしん=59.8% ずつき=11.5%
モデル出力=じしん 正解=ずつき
```

じしんはパラセクトに対してタイプ相性が $1/4$ なので間違っていると考えられます。

▼リスト 2.9　モデル出力の例 3

```
自分 マリルリ 90/226
れいとうビーム はかいこうせん バブルこうせん どくどく
相手 ラフレシア 11/198
=> れいとうビーム=84.2% はかいこうせん=0.3% バブルこうせん=15.2% どくどく=0.3%
モデル出力=れいとうビーム 正解=どくどく
```

れいとうビームが正しくて、正解データのどくどくは明らかに間違っています。ラフレシアはどくタイプを含んでいてどくどくは効果がありません。正解データも疑似的なものであり必ずしも正しくないため、これに対して正解率 100% を目指すのは得策ではないことを示しています。

モデルの定量的な正解率は 58.5% でしたが、正解データ自体が間違っている場合もあり、定性的には妥当な出力ができているように思われます。

本章では、どのパーティに対しても行動選択が行える単一のモデル「汎用行動選択モデル」を学習させる足掛かりとして、既存のパーティ固有行動選択モデルによって生成した行動選択結果を用いて教師あり学習することを試み、かなりパラメータ数の少ないモデルでなければ過学習が起きやすいことがわかりました。定量的にもっとも良かったモデルについて、定性的にある程度適切に動作していることが確認できました。今回の学習データ生成には計算時間が 1 日ほどかかっており、教師あり学習と強化学習では最適化の容易さが異なるとはいえ、1 日程度で生成できる規模のデータに対するモデルの適切な規模がわかりましたので、これをもとに次の章では強化学習を行います。

第 3 章

汎用行動選択モデルの強化学習

第 2 章では教師あり学習であらゆるパーティの行動選択を行える汎用行動選択モデルを学習させ、適切なモデル構造・規模を確認しました。本章では、同一構造のモデルに対し、バトルの勝敗を報酬とした強化学習に取り組みます。

3.1 強化学習システムの独自実装

実装面の課題として、まず強化学習フレームワークを選定する必要があります。もともと深層学習フレームワーク Chainer ベースの ChainerRL を使っていたのですが、Chainer の開発終了に伴い深層学習フレームワークを PyTorch に移行を進めているという背景があり、強化学習フレームワークも PyTorch に対応したものが必要です。フレームワークの候補はいくつかあり RLlib[*1]を少し触ってみたのですが、基本的に一人用ゲームの環境を想定した作りとなっており、対戦ゲームでモデル同士を自己対戦させるにはそれなりの追加実装を必要とします。また、状態によって選択可能な行動が制約される（技の PP 切れ、交代先が瀕死かどうかなど）ことに対応する実装も必要です。そこで今回は、既存の強化学習フレームワークを使わず自前で強化学習アルゴリズムの 1 つである DQN (Deep Q-Network)[*2]を実装することにしました。メリットは、ポケモンバトル用に特化した実装ができるため自由度が高く、コードの見通しが良くなります。デメリットは DQN 以外の強化学習アルゴリズム (A3C, ACER, PPO 等) を使いたくなった場合には各アルゴリズムを自前で実装しなければならない点です。本当は確率的な行動選択をモデリングする ACER 等の方策ベースの強化学習アルゴリズムのほうが読み合いを要するゲームにふさわしいように思えますが、その域に至るのは当分先と考え、実装が容易な

[*1] `https://docs.ray.io/en/master/rllib.html`

[*2] Volodymyr Mnih et al., Human-level control through deep reinforcement learning. Nature 518, 529-533, 2015.

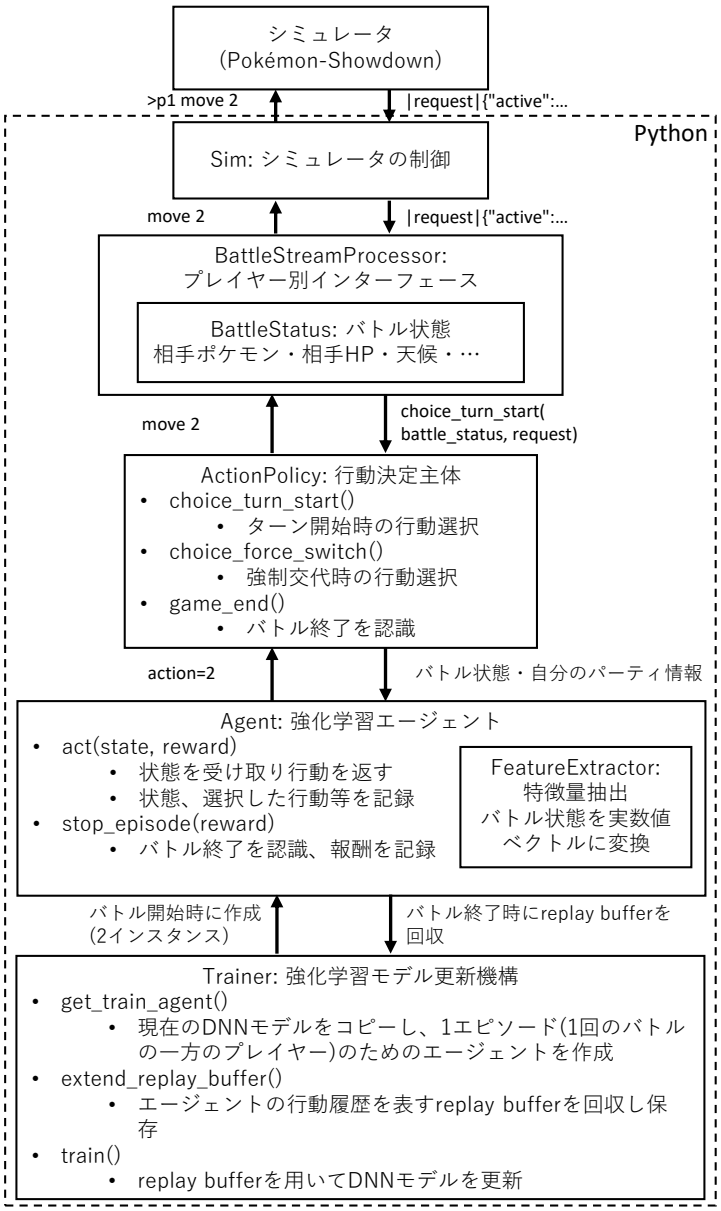
価値ベースの DQN を実装することにしました。より具体的には、DQN を少し改良した Double DQN[*3]を実装しました。DQN そのもののアルゴリズムについては参考書が多数あるため、省略させていただきます。

実装は、「リバーシ AI を作って学ぶ深層強化学習 (山岡忠夫)」 [*4]、「つくりながら学ぶ！ 深層強化学習 PyTorch による実践プログラミング (小川雄太郎)」を参考に DQN のコアアルゴリズムを実装しつつ、ポケモンバトルのシミュレータの呼出し方法などに即してデータの取り回しを独自に実装しました。

システムの構成を図 3.1 に示します。ポイントは、モデルの更新を行う `Trainer` と、バトル中の行動を決定する `Agent` を分離している点です。DQN の学習では、(`state`(ターンNの状態), `action`(ターンNで選択した行動), `next_state`(ターンN+1の状態), `reward`(ターンNで得た報酬)) という組の情報を replay buffer に集積してモデルの更新に使うわけですが、バトルには 2 人のプレイヤーが必要で、それぞれ同じターンに異なる状態を観測することになります（自分のポケモンが覚えている技の情報が `state` の一部に含まれるため)。`Agent` はプレイヤー 1 人分の視点で時系列を記録することにより見通しを良くしたうえで、両方のプレイヤーが得た replay buffer を `Trainer` が持つ単一の replay buffer に集積することで 1 つのモデルを更新するという仕組みとしました。

[*3] Hado Van Hasselt, Arthur Guez et al., Deep Reinforcement Learning with Double Q-Learning. In Proceedings of AAAI, 2016.

[*4] `https://tadaoyamaoka.booth.pm/items/1830557`



▲図 3.1　強化学習システムの構成

3.2 学習条件

強化学習関係のデフォルトの学習条件は以下のように設定しました。

- アルゴリズム: DQN (Double DQN)
- 探索: ϵ (epsilon)-greedy
  - ランダム行動する確率 ϵ: 0.3 (定数)
- 報酬割引率 γ （遠い将来に受け取る報酬を小さくするための係数）: 0.95
- バッチサイズ: 32
- 最初に学習するまでのステップ数 (replay buffer のサンプル数): 500
- N サンプル収集するたびにモデル更新: N=1
- N 回モデル更新するごとに target network を更新: N=100
- replay buffer サイズ: 100,000
- モデル: 3 層 16 チャンネル（教師あり学習と同一構造）
- バトル数: 10,000
  - バトルごとに、事前生成した 100 パーティから 2 パーティをランダムに選択し自己対戦
- 報酬: バトル終了ターンに、勝ちが +1、負けが-1

3.3 教師あり学習との比較

教師あり学習したモデルを強化学習エージェントと同じデータ形式に変換し、強さを比較します。強化学習エージェント、教師あり学習エージェントおよびランダムに行動するエージェントを混合してレーティングバトルを行いました。

第 3 巻までと異なり、エージェントは特定のパーティと一対一対応していません。エージェントとそれが操作するパーティの組み合わせをプレイヤーと呼ぶことにし、まずプレイヤーの強さを測ります。そして、あるエージェントを含むプレイヤーの強さの平均をエージェントの良さと定義します。ここでは、ランダムに生成したパーティ群とエージェントの全組み合わせをプレイヤーとして用います。例えばエージェント X,Y,Z とパーティ A,B があるとき、プレイヤー 1 はエージェント X がパーティ A を操作、プレイヤー 2 はエージェント X がパーティ B を操作、プレイヤー 3 はエージェント Y がパーティ A を操作、というように 6 人のプレイヤーがバトルに参加します。エージェント X の強さは、プレイヤー 1 の強さとプレイヤー 2 の強さを平均したものとなります。強さの定量化は、イロレーティングにより行います。イロレーティングは平均 1500 で、大きいほど

プレイヤーが強いことを示します。プレイヤー間のレート差が 200 のとき、上位のプレイヤーの勝率がおよそ 76% であることを示します。具体的な計算は、プレイヤー同士を様々な組み合わせで対戦させることで収束計算します。1 プレイヤーあたり 100 回対戦を行います。

まず強化学習の際に用いたのと同じパーティ群で評価します。3 エージェント、100 パーティの全組み合わせで合計 300 プレイヤーがバトルに参加することになります。表 3.1 に各エージェントが操作したプレイヤーの平均レートを示します。

▼表 3.1　教師あり学習と強化学習エージェントを対戦させ、各エージェントで操作したパーティの平均レート。

エージェント	平均レート
ランダム	1403
教師あり	1553
強化学習	1544

残念ながら、教師あり学習のほうが強いという結果になりました。教師あり学習は別のパーティ群で学習しているため不公平な面があるため、パーティ群を同じ条件で別途ランダムに生成して試した結果を表 3.2 に示します。

▼表 3.2　教師あり学習と強化学習エージェントを、学習に用いたのとは別に生成したパーティを操作して対戦させ、各エージェントで操作したパーティの平均レート。

エージェント	平均レート
ランダム	1422
教師あり	1565
強化学習	1513

さらに教師ありと強化学習の差が開きました。学習に用いていないパーティでもある程度操作できており汎化性能があることが示された一方、学習に使ったパーティのほうが他のパーティよりうまく操作できる傾向がみられます。

3.4　バトル内容の評価

強化学習エージェントだけで対戦させてレーティングおよびバトルログを観察します。

表 3.3・表 3.4 に最上位、最下位となったプレイヤーが用いていたパーティをそれぞれ 10 個示します。

▼表 3.3　強化学習エージェント同士の対戦でのレート上位 10 件のパーティ

レート	パーティ
1882	カビゴン, じしん, のしかかり, どろかけ, ソーラービーム
1843	ミルタンク, かいりき, じしん, でんじほう, のしかかり
1828	ファイヤー, おんがえし, かげぶんしん, だいもんじ, はかいこうせん
1744	マタドガス, かえんほうしゃ, はかいこうせん, かげぶんしん, ヘドロばくだん
1733	フリーザー, どろかけ, れいとうビーム, すなあらし, とっしん
1725	ハッサム, ロケットずつき, いわくだき, かげぶんしん, おんがえし
1704	ムウマ, おんがえし, スピードスター, サイコキネシス, かげぶんしん
1701	ギャラドス, いわくだき, ハイドロポンプ, なみのり, のしかかり
1699	フリーザー, ふぶき, かげぶんしん, はがねのつばさ, すてみタックル
1691	ケンタロス, すてみタックル, かえんほうしゃ, いわくだき, かいりき

▼表 3.4　強化学習エージェント同士の対戦でのレート下位 10 件のパーティ

レート	パーティ
931	エイパム, でんじほう, １０まんボルト, いわくだき, どくどく
1168	ヤンヤンマ, スピードスター, はがねのつばさ, ソーラービーム, つばさでうつ
1217	スピアー, ロケットずつき, ギガドレイン, スピードスター, はかいこうせん
1229	ダグトリオ, すてみタックル, とっしん, ヘドロばくだん, はかいこうせん
1249	バリヤード, どろかけ, ずつき, ロケットずつき, １０まんボルト
1266	ヤミカラス, かげぶんしん, つばさでうつ, スピードスター, ゴッドバード
1271	ヘルガー, かげぶんしん, どろかけ, いわくだき, スピードスター
1303	スピアー, どくどく, とっしん, ロケットずつき, ギガドレイン
1305	フーディン, ばくれつパンチ, とっしん, すてみタックル, おんがえし
1306	ルージュラ, すてみタックル, はなびらのまい, はかいこうせん, どろかけ

上位にはカビゴンをはじめとした有力なポケモンが来ています。あくまでランダム生成なので技構成が完璧というわけではありませんが、各ポケモンにそれなりにあった技を所持しています。最下位はエイパムでした。エイパムの特攻種族値は 40 で、特殊電気技は全くマッチしません。いわくだきも威力 20 で話になりません。ポケモンと技のミスマッチが生じているパーティが下位に来ていることが見て取れます。

最上位だったカビゴンのバトル中の行動を定性的に確認します。

▼リスト 3.1　カビゴンの対サンダース（電気タイプ）における行動選択

```
相手 サンダース HP187/187
自分 カビゴン HP292/292
じしん のしかかり どろかけ ソーラービーム
選択=> じしん

相手 サンダース HP52/187
自分 カビゴン HP207/292
じしん のしかかり どろかけ ソーラービーム
選択=> じしん
```

▼リスト 3.2　カビゴンの対ニョロトノ（水タイプ）における行動選択

```
相手 ニョロトノ HP215/215
自分 カビゴン HP292/292
じしん のしかかり どろかけ ソーラービーム
選択=> じしん

相手 ニョロトノ HP160/215
自分 カビゴン HP255/292
じしん のしかかり どろかけ ソーラービーム
選択=> じしん

相手 ニョロトノ HP100/215
自分 カビゴン HP177/292
じしん のしかかり どろかけ ソーラービーム
選択=> じしん

相手 ニョロトノ HP43/215
自分 カビゴン HP137/292
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり
```

原則にじしんを使い、相手の HP が減っているとのしかかりを使うのでしょうか？ タイプ一致を踏まえるとのしかかりのほうが威力が大きいため、本来はのしかかりをメインに使うのが正解と思われます。

▼リスト 3.3　カビゴンの対モンジャラ（草タイプ）における行動選択

```
相手 モンジャラ HP187/187
自分 カビゴン HP292/292
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり

相手 モンジャラ HP135/187
自分 カビゴン HP262/292
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり

相手 モンジャラ HP77/187 par
自分 カビゴン HP235/292
じしん のしかかり どろかけ ソーラービーム
```

```
選択=> のしかかり

相手 モンジャラ HP24/187 par
自分 カビゴン HP235/292
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり
```

じしんがいまひとつの相手（草タイプ）にはのしかかりを使っています。

▼リスト 3.4　カビゴンの対クロバット（毒・飛行タイプ）における行動選択

```
相手 クロバット HP209/209
自分 カビゴン HP292/292
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり

相手 クロバット HP135/209 par
自分 カビゴン HP239/292
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり

相手 クロバット HP67/209 par
自分 カビゴン HP239/292
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり

相手 クロバット HP67/209 par
自分 カビゴン HP239/292
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり
```

じしんが無効の相手（飛行タイプ）にものしかかりを使えています。

▼リスト 3.5　カビゴンの対ガラガラ（地面タイプ）における行動選択

```
相手 ガラガラ HP182/182
自分 カビゴン HP292/292
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり

相手 ガラガラ HP152/182
自分 カビゴン HP233/292 brn
じしん のしかかり どろかけ ソーラービーム
選択=> どろかけ

相手 ガラガラ HP147/182 accuracy-1
自分 カビゴン HP176/292 brn
じしん のしかかり どろかけ ソーラービーム
選択=> じしん

相手 ガラガラ HP125/182 accuracy-1
自分 カビゴン HP118/292 brn
じしん のしかかり どろかけ ソーラービーム
```

```
選択=> どろかけ

相手 ガラガラ HP120/182 accuracy-2
自分 カビゴン HP82/292 brn
じしん のしかかり どろかけ ソーラービーム
選択=> どろかけ

相手 ガラガラ HP115/182 accuracy-3
自分 カビゴン HP26/292 brn
じしん のしかかり どろかけ ソーラービーム
選択=> のしかかり
```

brn はやけど状態を表します。やけどしたところでなぜかどろかけを使うように変化します。間違った学習をしているようです。

技選択の正しさの簡易評価として、技の選択回数に対する効果抜群となった回数の比率を確認してみましたが、ランダムに行動するよりは高い値になっていることがわかりました。対戦組み合わせによってそもそも効果抜群の選択肢が存在するか否かが変動しますし、あまり安定する指標とは思えません。持続的に成果を確認できる定量評価指標が望まれます。

結論として、ある程度行動は正しいものの人が見て間違っている行動も混じっていました。定量的には教師あり学習より弱いという結果となっており、強化学習のパラメータの改善が必要と考えられます。

3.5 強化学習のハイパーパラメータチューニング

本節では、教師あり学習を超えるエージェントを強化学習によって学習することを目的とし、ハイパーパラメータチューニングを試みます。

予備実験で、バトル数 10,000 では不十分で、もっと多くのデータで学習すべきということが示唆されたため、バトル数は 100,000 に上げることにしました。学習中にランダムに行動する確率 ϵ は最初は大きく、学習が進むにつれて減らしたほうが良い[*5]という話もあります。バトル数を増やすにあたり、ステップ数に応じて ϵ を減少させることも試みます。具体的には、ϵ decay パラメータ d を追加し、ステップ数 s に対して $\epsilon(1-d)^s$ を ϵ として用いることにしました。例えば $d = 10^{-5}, s = 100000$ に対し $(1-d)^s = 0.368$ となり、ちょうどよい範囲の減少になります。すべてのハイパーパラメータを調整するには候補が多すぎるので、調整するハイパーパラメータの種類を絞ることとしました。バッチサイズなどは学習率の調整と近い効果を得られると考えられるので省略し、ϵ・ϵ decay・報

[*5] https://stackoverflow.com/questions/53198503/epsilon-and-learning-rate-decay-in-epsilon-greedy-q-learning

酬割引率・学習率の 4 つを調整することとしました。

調整結果の評価は、教師あり学習で得たモデルとの比較により定量化します。教師あり学習と強化学習エージェントに（いずれの学習にも用いていない）同一のパーティ群を操作させ、レーティングバトルでパーティ群の平均レートの差を計算します。

もっとも単純なパラメータの調整方法はグリッドサーチで、例えば ϵ の候補が $\{0.1, 0.5\}$、d の候補が $\{10^{-4}, 10^{-6}\}$ であれば、これらすべての組み合わせとして $(\epsilon, d) = \{(0.1, 10^{-4}), (0.1, 10^{-6}), (0.5, 10^{-4}), (0.5, 10^{-6})\}$ の 4 通りを試し、もっとも良かった結果をハイパーパラメータチューニングの結果として得ます。ただ、各変数を細かく調整しようとすると、「一変数当たりの候補数」の「変数の数」乗の組み合わせすべてをしらみつぶしに調べるため、非常にコストが大きくなります。今回だと 1 つのハイパーパラメータに対して強化学習を行うのに 5 時間程度かかるため、何百もの組み合わせを試すことができません。そこで、試す候補を減らしつつ、有望な結果を得るための手法を用いる必要があります。直観的な手順としては、まずランダムなハイパーパラメータ h_1, h_2 で学習を行い、評価 p_1、p_2（大きいほうが良いとする）を得ます。$p_1 > p_2$ なら h_1 のほうが有望であると考え、次に試すハイパーパラメータ h_3 は h_1 に近い値を選び、さらに評価が良くなるかを検証します。このように、過去の試行の結果を用いて有望なハイパーパラメータを選択するという手順を数学的に定式化した手法が提案されています。さらに、単に過去の履歴から評価値最大が期待されるハイパーパラメータだけでなく、過去の試行が偶然悪かった可能性も考慮したアルゴリズムとなっています。

3.5.1 ハイパーパラメータチューニングライブラリ Optuna の利用法

効率的にハイパーパラメータをチューニングしてくれるライブラリとして、今回は Python から扱いやすい Optuna[*6]を用いました。Optuna には前述のようにハイパーパラメータをチューニングするアルゴリズムが複数実装されており、簡単なインターフェースで利用することが可能です。今回使用するアルゴリズムはデフォルトの Tree-structured Parzen Estimator です。Optuna のバージョンは 2.0.0 です。

実験に使用したソースコードの Optuna 利用部分をかいつまんで具体的な利用方法を説明します。まず、`main` 関数で試行結果を保存するデータベースを開きます。Optuna は複数台のコンピュータを用いた並列探索に対応している点が特徴の 1 つで、データベースに試行結果（ハイパーパラメータとその評価結果）を保存します。今回はコンピュータ 1 台しか使いませんが、マルチコアを活かすため同時に複数の試行を走らせます。

[*6] `https://preferred.jp/ja/projects/optuna/`

▼リスト 3.6　Optuna で使用するデータベースを開く

```
import optuna
study = optuna.load_study(study_name="study1", storage="sqlite://study1.db")
```

storage 引数にはデータベースのアドレスを指定しますが、コンピュータ 1 台だけであれば sqlite を用いるとサーバソフトのインストール・常駐が不要で便利です。sqlite://study1.db は、カレントディレクトリの study1.db というデータベースファイルを使うことを示します。study_name はデータベース内で独立したハイパーパラメータチューニングのセッションを区別する名前です。データベースを作成するには optuna コマンドを用います。例えば

▼リスト 3.7　Optuna で使用するデータベースをコマンドラインから作成する

```
optuna create-study --study-name study1 --storage sqlite://study1.db
```

のように実行します。なお、optuna.create_study で Python コード内でデータベースを作成することもできます。

以下のコードで探索を開始します。

▼リスト 3.8　Optuna で探索を開始する

```
study.optimize(objective, n_trials=10)
```

objective は最適化対象の関数で、後述します。n_trials は試行回数です。今回は使用していませんが、n_jobs を指定すると、その数のプロセスが立ち上がって並列探索されます。この機能を使わずに複数のプロセスで同時に同じコードを実行した場合でも、データベースを介して連携することにより並列探索が可能となります。複数台のコンピュータを連携させる場合にも共通のデータベースにさえ接続できれば OS やネットワークの接続形態を問わないので非常にシンプルです。

objective 関数に、ハイパーパラメータを受け取って評価する機能を実装します。ちょっと長いのでコア部分だけ示します。

▼リスト 3.9　Optuna で最適化する対象の関数の実装

```
def objective(trial):
    trainer_id = ObjectId()
    # ハイパーパラメータを取得し、学習に必要な設定ファイルを作成
    param_file = make_train_param_file(
```

```
        epsilon=trial.suggest_uniform("epsilon", 0.1, 0.5),
        epsilon_decay=trial.suggest_loguniform("epsilon_decay", 1e-7, 1e-5),
        gamma=trial.suggest_uniform("gamma", 0.8, 1.0),
        lr=trial.suggest_loguniform("lr", 1e-4, 1e-1))
    # train_paramの値を用いて学習・評価する独自のコード
    subprocess.check_call(
        ["python", "-m", "pokeai.ai.generic_move_model.rl_train", param_file, ...
    ])
    rate_id = ObjectId()
    subprocess.check_call(
        ["python", "-m", "pokeai.ai.generic_move_model.rl_rating_battle", ...
    ])
    rate_advantage = get_rate_advantage(rate_id, ...)
    trial.set_user_attr("trainer_id", str(trainer_id))  # trialに追加情報を保存
    return -rate_advantage  # rate_advantageを最大化=Optunaでは最小化
```

objective 関数は trial という引数を受け取ります。この引数が Optuna が持っている試行の状態を表します。今回の objective の呼出しで試行すべきハイパーパラメータは、trial.suggest_uniform(name, low, high) で得ます。suggest_uniform の引数で、パラメータの名称、最小値、最大値を指定します。これは一様分布ですが、log スケールに対して一様分布を生成するには suggest_loguniform を用います。Optuna では、探索前にハイパーパラメータの数や範囲を指定するのではなく、実行中に指定するという形態をとっています。この機能を使うと、あるハイパーパラメータに依存して別のハイパーパラメータを生成させるようなことが可能です。今回は使用しませんが、公式マニュアルから例を引用します[*7]。

▼リスト 3.10　Optuna でハイパーパラメータ範囲を動的に変更する例

```
def objective(trial):
    classifier_name = trial.suggest_categorical('classifier',
                                                ['SVC', 'RandomForest'])
    if classifier_name == 'SVC':
        svc_c = trial.suggest_loguniform('svc_c', 1e-10, 1e10)
        classifier_obj = sklearn.svm.SVC(C=svc_c)
    else:
        rf_max_depth = int(trial.suggest_loguniform('rf_max_depth', 2, 32))
        classifier_obj = sklearn.ensemble.RandomForestClassifier(
            max_depth=rf_max_depth)
```

以下は、得られたハイパーパラメータを用いて強化学習と評価を行っています。objective は通常の Python コードですので、タスクごとに都合のいい手段で実装することができます。今回は、学習・評価コードを外部プロセス呼び出しで実行し、その結果を読み取って rate_advantage（実数値）に代入しています。Optuna は objective の戻り値

[*7] https://optuna.readthedocs.io/en/stable/tutorial/configurations.html

を最小化するように動作するため、最大化したい rate_advantage の符号を反転して返します。選ばれたハイパーパラメータと戻り値はデータベースに保存され、以後の探索に利用されます。今回は単に最適なハイパーパラメータを知るだけでなく、学習したモデルを後で使いたいので、モデルの保存のためにランダム生成した ID を trial.set_user_attr("trainer_id", str(trainer_id)) で trial に紐づけています。この値もデータベースに保存されるので、探索後に使うことができます。

なお、今回強化学習システムは独自に実装したものですので学習の実行や評価結果の計算は独自に行っていますが、Tensorflow などの著名な機械学習ライブラリの典型的な利用であれば、optuna.integration 以下にライブラリ間の橋渡しを行ってくれる機能があり、より短いコードで実装が可能です。

3.5.2 ハイパーパラメータチューニングの実験結果

Optuna を用いた強化学習のハイパーパラメータチューニングの結果を示します。

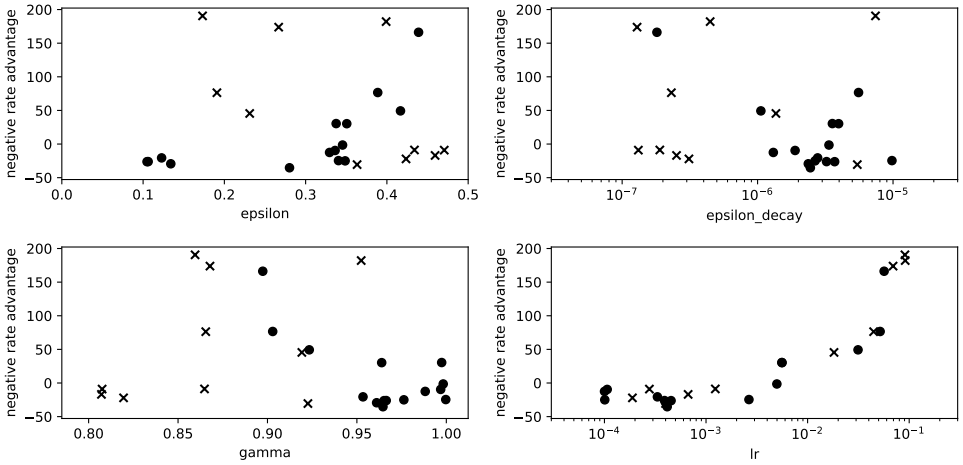
チューニング対象の変数と範囲は以下の通りです。

- ϵ: 0.1〜0.5
- ϵ decay: 10^{-7}〜10^{-5}
- 報酬割引率: 0.8〜1.0
- 学習率: 10^{-4}〜10^{-1}

25 回の試行を行った結果、各変数と評価（小さいほうが良く、0 未満で強化学習のほうが強いことを表す）の関係を図 3.2 に示します。

これらのグラフから、学習率が評価に大きな影響を与えていることが見て取れます。一方、他の変数については明確な傾向が見られません。複数の変数を同時に変化させながら試行しているため、横軸の値がほぼ同じでも別の変数の値が大きく異なり、縦軸の値には大きな変化が生じている場合があります。Optuna のアルゴリズムにより有望なパラメータの範囲内で多くの探索がなされるため、探索の後半では学習率が 4×10^{-4} 付近の試行回数が多くなっています。評価が最善だったのは、ϵ=0.28, ϵ decay=2.5×10^{-6}, 報酬割引率=0.96, 学習率=4.1×10^{-4} のときでした。そして評価結果は-35 で、これは同じモデル構造で教師あり学習よりも強いモデルが強化学習によって学習できたことを示します。

本章では、汎用行動選択モデルの強化学習を試み、ハイパーパラメータチューニングを行いました。結果としてパーティ固有行動選択モデルの行動から教師あり学習したエージェントより強いものができ、強化学習単体で汎用行動選択モデルを用いた行動決定が可能となりました。



▲図 3.2 各ハイパーパラメータと評価の関係。左上: ϵ、右上: ϵ decay、左下: 報酬割引率 (γ)、右下: 学習率 (lr)。最初の 10 試行を×、それ以降を●でプロット。

第 4 章

汎用行動選択モデルを用いたパーティ生成

第 3 章では、強化学習で汎用行動選択モデルの学習が可能だということを確認しました。本章では、学習結果のエージェントの様々な入力に対する出力を観察します。そして、バトル中の行動を決定するためのモデルが、強いパーティの生成にも活用できることを示します。

4.1 Q 関数の観察

強化学習アルゴリズムのひとつである DQN で学習されるモデルは Q 関数 $Q(s, a)$ と呼ばれます。Q 関数の出力を Q 値と呼びますが、単に各行動に対応する値の大小関係でどの行動を選ぶべきかを示すだけでなく、状態 s のときに行動 a をとったときに将来得られる報酬（厳密には割引報酬和）の期待値に対応します。ここで、報酬は勝ちが +1、負けが -1 に設定して学習してあります。つまり、勝ちが決定的な状態とその時選ぶべき行動の組み合わせに対しては +1 に近い値が得られ、勝敗が五分五分なら $+1 \times 50\% + (-1) \times 50\%$ で 0 に近い値が得られるように学習が進みます。うまく学習ができていれば、残り HP であったり相手との相性によって Q 値が変動する様子が観察できるはずです。

第 3 章で学習したエージェント[*1]同士で対戦させ、各局面での Q 値を記録しました。

まず、100 パーティ中レート 8 位のギャラドス（水・飛行タイプ）を使った場合の結果を示します。1 位のカビゴンは相性の影響がある場面が少なかったためです。

パラセクトを相手にしたバトルの全ターンを示します。技名の右のかっこ内の数値が Q

[*1] 実験順序の都合で、ハイパーパラメータチューニング前のモデルを用いています。具体的な数値には多少違いが生じますがご了承ください。

値を表します。

▼リスト 4.1　ギャラドスの対パラセクト（虫・草タイプ）における技ごとの Q 値

```
相手 パラセクト HP182/182
自分 ギャラドス HP220/220
いわくだき(0.31) ハイドロポンプ(0.24) なみのり(0.36) のしかかり(0.68)
選択=> のしかかり

相手 パラセクト HP127/182
自分 ギャラドス HP174/220
いわくだき(0.48) ハイドロポンプ(0.50) なみのり(0.55) のしかかり(0.75)
選択=> のしかかり

相手 パラセクト HP80/182 par
自分 ギャラドス HP174/220
いわくだき(0.85) ハイドロポンプ(0.85) なみのり(0.89) のしかかり(0.91)
選択=> のしかかり

相手 パラセクト HP32/182 par
自分 ギャラドス HP174/220
いわくだき(0.94) ハイドロポンプ(0.93) なみのり(0.97) のしかかり(1.00)
選択=> のしかかり
```

ハイドロポンプ等の水技が半減なので、のしかかりの Q 値が相対的に高くそれが実際の行動として選ばれています。そして、相手の HP が減り、自分の HP があまり減少していないという勝利に近い状況になると Q 値が大きくなっており、期待通りの結果になっているといえます。

次は天敵である電気タイプのサンダースとの対戦です。

▼リスト 4.2　ギャラドスの対サンダース（電気タイプ）における技ごとの Q 値

```
相手 サンダース HP187/187
自分 ギャラドス HP220/220
いわくだき(0.02) ハイドロポンプ(0.15) なみのり(0.15) のしかかり(0.22)
選択=> のしかかり

相手 サンダース HP130/187
自分 ギャラドス HP220/220
いわくだき(0.33) ハイドロポンプ(0.52) なみのり(0.46) のしかかり(0.51)
選択=> ハイドロポンプ

相手 サンダース HP64/187
自分 ギャラドス HP220/220
いわくだき(0.65) ハイドロポンプ(0.70) なみのり(0.72) のしかかり(0.71)
選択=> なみのり
```

最初のターンの時点で、パラセクト戦のときより Q 値が低いことがわかります。なお、自分のポケモンは種族を特徴量と入れており、相手のポケモンについては種族は入れずにタイプを入れています。タイプ相性による有利不利を判断できていると考えられます。

水技が抜群となるニドクインとの対面を示します。

▼リスト 4.3　ギャラドスの対ニドクイン（毒・地面タイプ）における技ごとの Q 値

```
相手 ニドクイン HP215/215
自分 ギャラドス HP220/220
いわくだき(-0.15) ハイドロポンプ(0.18) なみのり(0.37) のしかかり(0.31)
選択=> なみのり
```

水技の Q 値が高くなっており、技と相手の相性についても認識ができています。逆に、同じ局面をニドクイン側から見た場合を示します。

▼リスト 4.4　ニドクイン側から見た技ごとの Q 値

```
相手 ギャラドス HP220/220
自分 ニドクイン HP215/215
１０まんボルト(-0.25) だいもんじ(0.46) バブルこうせん(-0.37) ほのおのパンチ(0.34)
選択=> だいもんじ
```

逆にニドクイン側は最適な１０まんボルトの Q 値がかなり低くなっており、間違いもまだまだ多いです。このような間違いにより、この強化学習モデルは教師あり学習モデルより弱くなっていると考えられます。

次に、レート最下位のエイパムについての結果を示します。

電気技が有効なフリーザーとの対戦です。

▼リスト 4.5　エイパムの対フリーザー（氷・飛行タイプ）における技ごとの Q 値

```
相手 フリーザー HP215/215
自分 エイパム HP176/176
でんじほう(0.24) １０まんボルト(0.17) いわくだき(-0.07) どくどく(0.31)
選択=> どくどく

相手 フリーザー HP197/215 tox
自分 エイパム HP102/176
でんじほう(0.10) １０まんボルト(0.10) いわくだき(-0.03) どくどく(0.04)
選択=> １０まんボルト

相手 フリーザー HP123/215 tox
自分 エイパム HP31/176
でんじほう(-0.20) １０まんボルト(-0.29) いわくだき(-0.42) どくどく(-0.36)
選択=> でんじほう
```

まずどくどくで状態異常にしてから、電気技で攻めていくという戦略です。とはいえフリーザーの能力値が高いので勝てそうにないですが。ほかのバトルも観察すると、相手が状態異常でないときにどくどくを使うのではなく、自分の HP が減っていないときに使うという判断に見えました。

この技構成では手も足も出ないニドクインとの対面を示します。

▼リスト 4.6　エイパムの対ニドクインにおける技ごとの Q 値（バトル開始時）

```
相手 ニドクイン HP215/215
自分 エイパム HP176/176
でんじほう(-0.35) １０まんボルト(-0.78) いわくだき(-0.69) どくどく(-0.33)
選択=> どくどく
```

初手の時点ですでにすべての Q 値がマイナスです。

この対面の最終ターンです。

▼リスト 4.7　エイパムの対ニドクインにおける技ごとの Q 値（最終ターン）

```
相手 ニドクイン HP215/215
自分 エイパム HP5/176
でんじほう(-1.06) １０まんボルト(-1.16) いわくだき(-1.15) どくどく(-1.12)
選択=> でんじほう
```

負けが確定的な場面で、Q 値が-1 より低くなっています。

ここまでで、タイプ相性や HP の減り方によって Q 値が期待通り変動していることを確認できました。

さらに、相性の有利不利がほとんどない組み合わせにおける、最初のターンにおける出力を比較してみます。

▼リスト 4.8　ギャラドスの対リングマ（ノーマルタイプ）における技ごとの Q 値

```
相手 リングマ HP215/215
自分 ギャラドス HP220/220
いわくだき(-0.07) ハイドロポンプ(0.30) なみのり(-0.11) のしかかり(0.02)
選択=> ハイドロポンプ
```

▼リスト 4.9　エイパムの対リングマにおける技ごとの Q 値

```
相手 リングマ HP215/215
自分 エイパム HP176/176
でんじほう(-0.27) １０まんボルト(-0.58) いわくだき(-0.58) どくどく(-0.39)
選択=> でんじほう
```

自分のポケモンによって Q 値に大きな違いがあることがわかります。特にいわくだきに対する Q 値は、入力特徴としては自分のポケモンの種族だけが異なっている状態であり、ポケモンの強さを表現しているものと考えられます。またいわくだきは比較的弱い技

なので、他の技より低い Q 値となることが多いです。ここから、「バトルの最初のターンにおける Q 値を、パーティの良さを表す指標として使えるのではないか」という仮説を提唱します。パーティの良さを表す指標＝パーティ評価関数を用いて強いパーティを構築する手法は過去に提案しました。このときはバトル中の行動の強化学習とは別個にパーティ評価関数を学習していましたが、Q 関数の利用によりパーティの生成も実現できることが期待されます。

強化学習によって得た Q 関数を用いて、バトル中の様々な状況における Q 値を観察しました。タイプ相性や HP の減り方によって Q 値が期待通り変動していることを確認できました。さらに、バトルの最初のターンにおける Q 値はパーティの良さを表しているという仮説を立てました。次節で、Q 関数をパーティ生成に活用する手法を検討します。

4.2 Q 関数を用いたパーティの強さの定式化

強化学習を通じて学習されるエージェントの Q 関数 $Q(s, a)$ の s にはバトルの状態、a には選択する行動が代入されます。現在は交代なしの 1vs1 バトルを扱っており、バトル開始直後の s には相手のポケモンのタイプ（17 次元のうち該当する次元が 1）が含まれます。残り HP などの要素もありますが、定数になります。また、a には自分のポケモンの種族（251 次元のうち該当する次元が 1）および技（251 次元のうち該当する次元が 1）が含まれます。Q 値は 1 つの状況に対して選択肢となる技ごとに異なる a が与えられるため、異なる Q 値が得られます。相手ポケモンが変化すれば s が変化し、Q 値が変化します。

良いパーティは、どんな相手に対しても高い Q 値が得られるパーティであると考えられます。これを定式化すると、パーティ X に対してパーティの良さを表す関数 $R(X)$ は次のようになります。

$$R(X) = \frac{1}{|E|} \sum_{E} \max_{a} Q(s(E), a)$$

ここで、E は相手として想定するポケモンの集合、$s(E)$ は相手ポケモンを代入した状態ベクトルです。今回、E にはポケモン全種類（最終進化系 129 種類）を用いることにします。言葉で説明すれば、R は相手ポケモンに対して最善の技を選択したときの Q 値を、想定されるすべての相手ポケモンに対して平均したものとなります。平均ではなくて max なのは、相手ごとに有効な技を一つ覚えていれば十分であるためです。

具体例として、上に示したギャラドスのパーティに対する R は、$E = \{$ パラセクト, ニドクイン $\}$ としたときに

$$
\begin{aligned}
R(X) &= \frac{1}{2}(\max\{0.31, 0.24, 0.36, 0.68\} + \max\{-0.15, 0.18, 0.37, 0.31\}) \\
&= \frac{1}{2}(0.68 + 0.37) \\
&= 0.525
\end{aligned}
$$

となります。

4.3 Q 関数によるパーティ評価の実験

実際に上記の定式化を使って、ランダムに生成したパーティ 1,000 個の評価を行いました。

▼表 4.1 R の値上位 10 パーティ

R	パーティ
0.675	フォレトス, かげぶんしん, すてみタックル, ソーラービーム, ギガドレイン
0.665	サイドン, かいりき, だいもんじ, じしん, ふみつけ
0.664	フォレトス, かげぶんしん, いわくだき, ギガドレイン, スピードスター
0.656	ヤドラン, じしん, かげぶんしん, ちきゅうなげ, とっしん
0.651	ガルーラ, ほのおのパンチ, のしかかり, かえんほうしゃ, いわくだき
0.646	サイドン, じしん, ばくれつパンチ, かえんほうしゃ, バブルこうせん
0.646	フォレトス, ころがる, おんがえし, とっしん, スピードスター
0.636	フォレトス, ずつき, すてみタックル, スピードスター, すなあらし
0.634	フォレトス, いわくだき, ソーラービーム, ころがる, すてみタックル
0.634	フォレトス, かげぶんしん, おんがえし, どくどく, いわくだき

▼表 4.2 R の値下位 10 パーティ

R	パーティ
-0.601	レディアン, スピードスター, ばくれつパンチ, かげぶんしん, ずつき
-0.596	レディアン, ソーラービーム, すてみタックル, ずつき, ころがる
-0.584	レディアン, おんがえし, かげぶんしん, すてみタックル, ソーラービーム
-0.501	レディアン, どくどく, おんがえし, スピードスター, はかいこうせん
-0.501	レディアン, はかいこうせん, どくどく, おんがえし, すてみタックル
-0.496	レディアン, ばくれつパンチ, ずつき, れいとうパンチ, ソーラービーム
-0.383	ヤンヤンマ, かげぶんしん, ソーラービーム, ずつき, はがねのつばさ
-0.379	ヤンヤンマ, つばさでうつ, ずつき, ソーラービーム, はがねのつばさ
-0.378	ヤンヤンマ, スピードスター, つばさでうつ, ずつき, ソーラービーム
-0.366	ヤンヤンマ, はがねのつばさ, つばさでうつ, スピードスター, おんがえし

最上位、最下位のパーティは表 4.1・表 4.2 のようになりました。上位はフォレトスが占めています。あまり強いというイメージはないですが、物理技で突破するのは困難なので交代なしのルールでは強いのかもしれません。サイドンなども強力な技を覚えた個体が上位に来ていることが確認できます。下位はレディアンとヤンヤンマでした。この時代、タイプ一致の虫タイプも飛行タイプも技に恵まれていないので妥当な結果と思われます。

このように、Q 関数でパーティの強さを評価することができそうだと分かりました。

4.4 Q 関数による強いパーティの生成

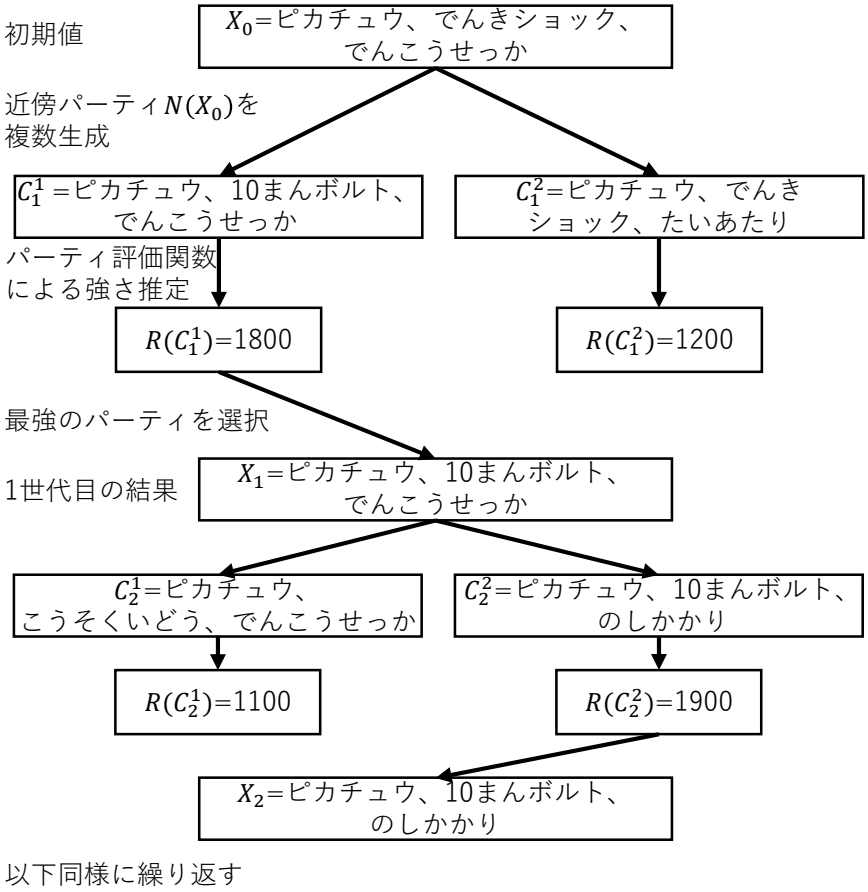
次に、既存のパーティを評価するだけでなく、強いパーティの生成を試みたいと思います。ここでの目的は、パーティ評価関数 $R(X)$ の値が比較的大きいパーティ X を多数生成することです。上位の目的は強化学習中に対戦させるパーティを生成することであり、強化学習エージェントは様々なパーティを受け持ち、様々な相手への立ち回り方を学習する必要があります。そのため R(X) が厳密な最大値をとる唯一のパーティを生成するのではなく、$R(X)$ をできるだけ大きくしつつ、パーティごとにポケモン・技が異なるような多様性のあるパーティ群 $X_G = \{X^1, X^2, X^3, ...\}$ を生成します。

山登り法を用いたパーティ生成手法について、第 3 巻から必要な部分を変更して再掲します。

パーティ評価関数 $R(X)$ の値が最大となるようなパーティ X を生成するために、単純な離散最適化手法である山登り法を用います。山登り法の概要を図 4.1 に示します。最初にランダムなパーティ X_0 を生成し、これを少しだけ変更した近傍パーティ候補 $N(X_0) = \{{C_1}^1, {C_1}^2, ...\}$ を生成します。そして、これを R で評価し、最も強いパーティを選択します。この処理を 1 世代とします。選択されたパーティを X_1 とし、次の世代では再度これを少しだけ変更したパーティ候補 $N(X_1) = \{{C_2}^1, {C_2}^2, ...\}$ を生成します。そして、これを R で評価し、最も強いパーティを選択します。これを繰り返すことで徐々に強いパーティへと変化させていきます。ここで、パーティを少しだけ変更する手段は 2 通り用意しています。(1) ポケモンを 1 匹完全に別のものにし、技もランダムに設定する、(2)1 匹のポケモンを選び、その技を 1 つ選び、別のものにする。(1) が選ばれる確率は 10%、それ以外では (2) が選ばれ、パーティが変更されます。

4.4.1 ペナルティ項つき山登り法による多様なパーティ生成

山登り法は過去の巻でも使用しています。第 1 巻では、パーティ構成 1 匹の条件で用いており、パーティ変更の手段 (1) を使いませんでした。ポケモンによって覚える技が違うため、ポケモンを別のものに変更すると技はランダムに設定しなおすこととなり、1 匹し



▲図 4.1　山登り法の概要

かいないポケモンを変更すると近傍ではなくなってしまうという考え方で変更を避けました。しかしその場合、初期値としてランダムに選んだポケモンが弱いと改善の余地がないという状態になります。現在の条件で強いとされているフォレトスやサイドンと、極端に弱いレディアンやヤンヤンマが同じ頻度で現れるというのは強いパーティの生成として不十分と考えられます。第 3 巻では、パーティ構成 3 匹の条件で用いて、ポケモン変更ありで強いパーティを得ました。今回はパーティ構成 1 匹で、単純に同様の手法をとると同一のパーティが大量にできてしまうという結果になりました。山登り法は局所最適解が出る手法なので、初期値によって違う結果がある程度得られますが、探索範囲が狭いと同じ解に収束する可能性が高くなります。

この問題に対して、すでに生成されたパーティと類似のパーティが生成されにくくするための明示的なペナルティ項 $S(X)$ を設けることにしました。パーティ群に含まれるパーティは、$X^1, X^2, ...$ の順に 1 つずつ山登り法で生成します。X^i を生成する際は、それまでに生成したパーティと類似するパーティのパーティ評価関数の値を小さくし、今までに生成されていないパーティの生成を促進します。ペナルティ項は

$S(X_i) = \frac{1}{i-1} \sum_{j=1}^{i-1} K(X_j, X_i)$

と定義します。ここで、$K(X_j, X_i)$ はパーティ間の類似度を表す関数です。過去に提案したパーティを表す特徴量 (P,M,PM,MM) の内積として定義しました。パーティ間でポケモン 1 匹、技 1 つ、ポケモンと技の組み合わせ、技と技の組み合わせがいくつ一致するかで算出されます。パーティ間で一切一致がない場合は 0、ポケモンも技もすべて一致する場合は 1+4+4+6(4 つの技から 2 つを選ぶ組み合わせ)=15 になります。

このペナルティ項を用いて、山登り法でパーティを評価する際の計算式を $R(X_i) - \lambda S(X_i)$ とします。ペナルティの強さは負でないスカラー定数 λ で制御します。

4.5 パーティ生成実験

ペナルティの強さを変えつつパーティを生成して比較します。

山登り法のパラメータは、近傍パーティ生成数 10、世代数 100 とし、パーティ群の大きさは 871 としました。871 という中途半端な値の意味は第 5 章で説明します。

$\lambda = \{0, 0.1, 1, 10\}$ で生成した 871 パーティからランダムに 10 パーティ抽出した結果を示します。

▼リスト 4.10　$\lambda = 0$ で生成したパーティの例

```
ヤドラン,じしん,すてみタックル,だいもんじ,ふみつけ
ガルーラ,かえんほうしゃ,ちきゅうなげ,かみなり,だいもんじ
ヤドラン,じしん,だいもんじ,のしかかり,でんじほう
サイドン,だいもんじ,じしん,でんじほう,１０まんボルト
フォレトス,ころがる,すてみタックル,とっしん,ギガドレイン
ガルーラ,だいもんじ,かえんほうしゃ,かみなり,ちきゅうなげ
サイドン,じしん,１０まんボルト,だいもんじ,でんじほう
サイドン,だいもんじ,１０まんボルト,じしん,でんじほう
ガルーラ,かみなり,かえんほうしゃ,だいもんじ,ちきゅうなげ
フォレトス,とっしん,ころがる,ギガドレイン,すてみタックル
```

▼リスト 4.11　$\lambda = 0.1$ で生成したパーティの例

```
スイクン,れいとうビーム,なみのり,いわくだき,たきのぼり
ガルーラ,れいとうパンチ,だいもんじ,いわくだき,かみなり
プテラ,だいもんじ,はがねのつばさ,げんしのちから,じしん
```

```
エアームド,そらをとぶ,ゴッドバード,すなあらし,どろかけ
フォレトス,とっしん,ころがる,ギガドレイン,ソーラービーム
フォレトス,かいりき,ころがる,おんがえし,ギガドレイン
サイドン,ふぶき,じしん,でんじほう,ほのおのパンチ
ケンタロス,かみなり,ふみつけ,つのドリル,すてみタックル
フォレトス,ソーラービーム,かいりき,すてみタックル,ギガドレイン
ガルーラ,なみのり,かえんほうしゃ,かみなり,ちきゅうなげ
```

▼リスト 4.12　$\lambda = 1$ で生成したパーティの例

```
ドードリオ,のしかかり,どくどく,ゴッドバード,はがねのつばさ
ミルタンク,かみなりパンチ,ふみつけ,ちきゅうなげ,ずつき
カイリキー,かいりき,ほのおのパンチ,すてみタックル,ちきゅうなげ
ツボツボ,じしん,ヘドロばくだん,いわくだき,ころがる
カメックス,ころがる,たきのぼり,ハイドロポンプ,れいとうパンチ
フーディン,ほのおのパンチ,れいとうパンチ,サイケこうせん,かみなりパンチ
エアームド,はがねのつばさ,そらをとぶ,スピードスター,すなあらし
ラプラス,サイコキネシス,はかいこうせん,つのドリル,いわくだき
サイドン,つのドリル,ころがる,どろかけ,かみなり
エアームド,おんがえし,はがねのつばさ,ゴッドバード,どくどく
```

▼リスト 4.13　$\lambda = 10$ で生成したパーティの例

```
マリルリ,ふぶき,ハイドロポンプ,ばくれつパンチ,ずつき
プテラ,げんしのちから,ゴッドバード,はがねのつばさ,とっしん
ライコウ,すなあらし,いあいぎり,かみなり,スピードスター
ハガネール,いあいぎり,じしん,はかいこうせん,ロケットずつき
スイクン,バブルこうせん,いわくだき,スピードスター,なみのり
スイクン,たきのぼり,いあいぎり,なみのり,かげぶんしん
ピクシー,はなびらのまい,すてみタックル,ふぶき,ほのおのパンチ
ゴルダック,ハイドロポンプ,スピードスター,はなびらのまい,れいとうパンチ
デンリュウ,かみなりパンチ,とっしん,かみなり,いわくだき
ガルーラ,すなあらし,かえんほうしゃ,ほのおのパンチ,はかいこうせん
```

各々のパーティの良さは直観的にはどの条件でもあまり変わらない一方、$\lambda = 0$ ではポケモンの種類が偏っています。実際、「フォレトス, ころがる, すてみタックル, とっしん, ギガドレイン」のパーティは 219 回も生成されてしまいました。別の側面では、ランダムな初期値から開始して同じ解に収束しているので、山登り法の世代数などのパラメータは十分であると考えられます。λ が大きくなるにつれてパーティの多様性が増していますが、直観的に異常な（極端に弱そうな）内容にはなっていません。

▼表 4.3　λ に対する生成パーティのポケモンの頻度（上位 20 件）

$\lambda = 0$	$\lambda = 0.1$	$\lambda = 1$	$\lambda = 10$
フォレトス (223)	フォレトス (159)	ガルーラ (72)	サイドン (21)
ガルーラ (162)	ガルーラ (150)	フォレトス (63)	ガルーラ (19)
ケンタロス (91)	ヤドラン (99)	サイドン (62)	ケンタロス (19)
サイドン (86)	サイドン (91)	ケンタロス (62)	ミルタンク (17)
ヤドラン (85)	ミルタンク (81)	ミルタンク (58)	プテラ (17)
ミルタンク (68)	ケンタロス (73)	ヤドラン (51)	フォレトス (16)
エアームド (50)	エアームド (67)	エアームド (48)	ヤドラン (16)
スイクン (38)	スイクン (53)	スイクン (43)	エアームド (16)
ラプラス (29)	ラプラス (51)	ラプラス (42)	ラプラス (16)
プテラ (29)	プテラ (31)	プテラ (38)	メガニウム (16)
マンタイン (2)	カビゴン (6)	ハッサム (29)	ドードリオ (15)
カイリキー (2)	ハッサム (4)	ドードリオ (26)	ゲンガー (15)
ハッサム (2)	レアコイル (3)	カビゴン (23)	フシギバナ (14)
レアコイル (2)	ランターン (1)	マンタイン (18)	カビゴン (14)
ツボツボ (1)	カイリキー (1)	カイリキー (18)	カイリュー (14)
カビゴン (1)	ドードリオ (1)	フーディン (17)	ハッサム (14)
-	-	レアコイル (17)	スイクン (12)
-	-	メガニウム (12)	ヤミカラス (12)
-	-	ツボツボ (11)	マンタイン (12)
-	-	ゴルダック (10)	ネイティオ (12)

▼表 4.4　λ に対する生成パーティの技の頻度（上位 20 件）

$\lambda = 0$	$\lambda = 0.1$	$\lambda = 1$	$\lambda = 10$
だいもんじ (360)	じしん (223)	じしん (114)	じしん (79)
かみなり (287)	かみなり (179)	かみなり (97)	おんがえし (76)
じしん (271)	ギガドレイン (144)	だいもんじ (93)	れいとうビーム (75)
すてみタックル (260)	れいとうビーム (138)	おんがえし (89)	どくどく (75)
かえんほうしゃ (252)	だいもんじ (126)	れいとうビーム (89)	だいもんじ (74)
ギガドレイン (223)	おんがえし (112)	のしかかり (83)	かげぶんしん (74)
とっしん (222)	すなあらし (106)	サイコキネシス (83)	かみなり (73)
ころがる (221)	かえんほうしゃ (103)	すなあらし (81)	のしかかり (73)
ちきゅうなげ (220)	すてみタックル (98)	どくどく (81)	サイコキネシス (73)
のしかかり (148)	ちきゅうなげ (96)	すてみタックル (79)	スピードスター (73)
れいとうビーム (140)	なみのり (89)	はかいこうせん (79)	すてみタックル (72)
でんじほう (139)	のしかかり (88)	かげぶんしん (77)	とっしん (72)
ふみつけ (119)	１０まんボルト (87)	ずつき (77)	はかいこうせん (72)
１０まんボルト (85)	とっしん (83)	でんじほう (76)	ずつき (72)
すなあらし (82)	かみなりパンチ (83)	かえんほうしゃ (75)	どろかけ (72)
かみなりパンチ (71)	はかいこうせん (82)	とっしん (75)	いわくだき (72)
おんがえし (61)	かいりき (80)	かいりき (75)	すなあらし (71)
そらをとぶ (50)	でんじほう (79)	なみのり (75)	ロケットずつき (71)
ドリルくちばし (50)	れいとうパンチ (79)	いわくだき (75)	ハイドロポンプ (71)
はかいこうせん (41)	ずつき (79)	ハイドロポンプ (74)	ギガドレイン (70)

定量的に、全パーティでのポケモンや技の出現回数を算出した結果を表 4.3・表 4.4 に示します。ペナルティ項が大きくなるほど出現回数の偏りが小さくなっていることが確認できます。$\lambda = 0$ では最大出現回数のポケモンはフォレトスでしたが $\lambda = 1$ ではガルーラに変わっています。これは、フォレトスよりガルーラのほうが覚える技の種類が多いため、フォレトスを含むパーティ同士よりガルーラを含むパーティ同士のほうが類似度を抑えつつ多く生成できたものと考えられます。$\lambda = 10$ まで大きくすると、分布が平準化されすぎて強さの評価が失われていると考えられます。分布の広がりの必要性は、生成されたパーティ群を用いた行動の強化学習において様々な相手と対戦し偏りが出ないようにするという要請から来たものであり、単体でその有効性を定量評価することは困難です。$\lambda = 1$ の場合に生成されたパーティが定性的に悪くないので、今後はこのパラメータでパーティ生成をすることとします。

本章では、バトル中の行動選択のために学習した強化学習エージェントの Q 関数の入力として自分のパーティ構成を与えることでその強さが予測できることを発見しました。そして Q 値が最大となるようなパーティ構成を探索することで、バトルの前段階となる

強いパーティを生成する手法を提案しました。1 つのモデルでポケモンバトル全体のパイプラインをカバーできる、シンプルな手法が実現できました。

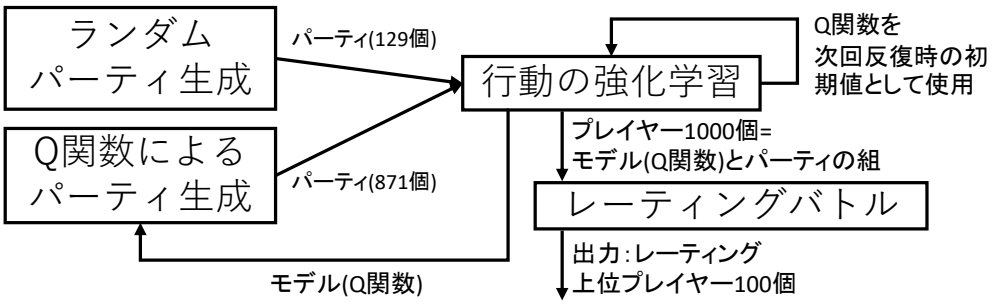
第 5 章

パーティ生成と行動選択の交互学習

第 3 章では、強化学習のハイパーパラメータを調整し、疑似教師データを使った教師あり学習よりも強い汎用行動選択モデルを学習させることに成功しました。これと、第 4 章で提案した汎用行動選択モデルを用いたパーティ生成手法を組み合わせて強いパーティとその適切な運用法を学習します。

学習システムの構成を図 5.1 に示します。

目的を単純に言えば、人間同士の対戦界において、ある戦略が普及すればそれに対抗するメタ戦略が考案され、戦略が進歩していくのと同様のことを AI の中で実現しようとしています。このシステムでは、パーティ生成と強化学習を交互に反復して行います。最初の反復 (t=0) では、Q 関数がないのでランダムにパーティを 1000 個生成します。次に、このパーティ群上でバトル中の行動選択モデルを強化学習により学習します。ここで学習されたエージェントと、対象にしたパーティ群の組み合わせをプレイヤー群とし、レーティングバトルによって上位のプレイヤー 100 個をその反復で生成されたプレイヤーとして出力します。次の反復 (t=1) では、t=0 で学習した行動選択モデルの Q 関数を用いて 871 個のパーティを生成し、これとランダムなパーティ 129 個（全最終進化系ポケモン 1 匹ずつ、技はランダム）を合わせて 1000 パーティを対象に強化学習を行います。強化学習モデルのパラメータは、前回の反復で学習したものを初期値として用います。以上の手順を反復し、洗練されたプレイヤーを生成します。最終的に、全反復の出力プレイヤーを混合してレーティングバトルさせ、上位のプレイヤーをシステム全体の出力とします。第 3 巻では、ランダムなパーティにランダムな行動させた結果を用いてパーティを生成し、強化学習するだけで終わっていました。今回の手法により、(1) 強化学習によりパーティを適切に運用したときに効果が大きい技を次の反復でのパーティ生成で優先的に組み込むことができるため運用法が難しい技を含めた最適なパーティを生成すること、(2) ある反復で強いとされたパーティに対する対抗手段を学習することで弱点の少ない運用法を学習することを目指します。



▲図 5.1　パーティ生成と強化学習を交互に行うシステムの構成

5.1　実験条件

実験条件は以下の通りです。

- 反復回数: 10
- パーティ数
  - Q 関数を用いて生成するパーティ数: 871
  - ランダム生成するパーティ数: 129
- パーティの生成条件
  - ポケモンの制限: 最終進化系 129 種類（ミュウツー・ドーブル等除く）
  - 技の制限: 効果が見込める 52 種類（高威力攻撃技主体）
  - 類似パーティ生成抑制のペナルティ λ: 1
- 強化学習
  - 探索: ϵ-greedy
    - ランダム行動する確率 ϵ: 0.3
    - ϵ decay: 2.0×10^{-6}
  - 報酬割引率: 0.95
  - optimizer (Adam) の学習率: 0.0004
  - バトル数: 100,000

強化学習のハイパーパラメータは、第 3 章で最適化したものを丸めました。記載のないものは第 3 章と同じです。

5.2 各反復の結果

各反復ではパーティが 1,000 個生成されます。それらに含まれるポケモン・技の頻度（それぞれ出現回数トップ 10）を表 5.1・表 5.2 に示します。ランダム生成結果そのままである反復 0 では、最初はランダムなので、ポケモンは均等に出現、技は覚えるポケモンの数が多いどくどく・かげぶんしんなどが上位に来ています。反復 1 では、強化学習結果からパーティ生成を経ることで、強そうなポケモンが上位に来ています。反復 2 では、ポケモン金銀最強と言われたカビゴンがトップとなりました。カイリューはトップ 10 から消えています。この理由を推察すると、反復 0 の時点では氷タイプの強力な技を持つポケモンが少なかったため上位に来て、その後氷タイプの技で対策すればあまり強くないということが学習された可能性があります。反復 2 以降、反復によって若干順位が入れ替わりますが、カビゴンは不動の 1 位でした。技はサンダー・ライコウに欠かせない１０まんボルト（かみなりも選択しうる）が 1 位となりました。

▼表 5.1　反復ごとの生成パーティ内のポケモン頻度

反復 0(ランダム生成)	反復 1	反復 2	反復 9(最終)
ピクシー (14)	カイリュー (87)	カビゴン (99)	カビゴン (104)
ツボツボ (12)	ミルタンク (63)	サンダー (76)	サンダー (83)
マタドガス (12)	ケンタロス (45)	ハピナス (64)	キングドラ (66)
ライコウ (12)	サンダー (43)	スターミー (51)	ライコウ (47)
モルフォン (11)	ゲンガー (39)	ガルーラ (51)	ガルーラ (46)
ピジョット (11)	ファイヤー (38)	マンタイン (41)	メガニウム (43)
キレイハナ (11)	ガルーラ (38)	ケンタロス (41)	デンリュウ (39)
ゴローニャ (11)	メガニウム (36)	ヤドキング (39)	ベトベトン (34)
エアームド (11)	キングドラ (33)	サンダース (38)	バリヤード (34)
トゲチック (11)	カビゴン (29)	ライコウ (29)	ラプラス (30)

▼表 5.2　反復ごとの生成パーティ内の技頻度

反復 0(ランダム生成)	反復 1	反復 2	反復 9(最終)
どくどく (246)	１０まんボルト (121)	おんがえし (129)	１０まんボルト (126)
かげぶんしん (229)	じしん (113)	１０まんボルト (113)	おんがえし (115)
おんがえし (219)	なみのり (102)	サイコキネシス (104)	なみのり (114)
はかいこうせん (170)	れいとうビーム (98)	かみなり (103)	かみなり (108)
とっしん (161)	だいもんじ (94)	すてみタックル (102)	れいとうビーム (100)
ずつき (143)	おんがえし (91)	じしん (101)	じしん (98)
どろかけ (138)	かえんほうしゃ (89)	れいとうビーム (99)	サイコキネシス (95)
すてみタックル (137)	ハイドロポンプ (87)	のしかかり (98)	どくどく (92)
スピードスター (137)	サイコキネシス (83)	なみのり (92)	のしかかり (91)
いわくだき (115)	かみなり (83)	ハイドロポンプ (90)	ふぶき (88)

各反復におけるパーティとエージェントを組み合わせたプレイヤー 1,000 個をレーティングバトルさせ、上位 10 プレイヤーを示します。反復 0（表 5.3）では、最初の時点でカビゴン・サンダーが強いということは学習の結果として現れています。反復 1（表 5.4）では、カビゴンのパーティへの採用頻度はまだ最大ではありませんが、すでに上位をカビゴンがほとんど占めています。反復 2（表 5.4）では、つのドリルを覚えたラプラス、ケンタロスが登場しています。最終の反復 9（表 5.4）では、「THE カビゴン 無双」といった状態です。ただし、最上位のカビゴンでもかえんほうしゃやだいもんじではなくほのおのパンチを覚えている（当時はすべて特殊技、ほのおのパンチのメリットはない）ので、技構成がパーフェクトかというとまだ疑問が残ります。とはいえ対戦の結果なので、これで差がつく場面はほとんどなかったという結果だと考えられます。

▼表 5.3　反復 0 内での対戦トップ 10 プレイヤーのパーティ

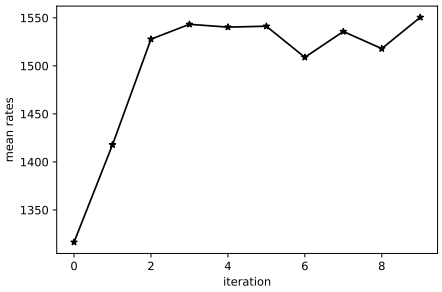
レート	パーティ
1997	カビゴン, おんがえし, はかいこうせん, ほのおのパンチ, すなあらし
1935	サンダー, はかいこうせん, ドリルくちばし, どろかけ, １０まんボルト
1917	サンダー, おんがえし, かみなり, いわくだき, すなあらし
1896	ライコウ, どろかけ, かいりき, １０まんボルト, かみなり
1892	ライコウ, ずつき, １０まんボルト, すなあらし, でんじほう
1891	リングマ, ばくれつパンチ, いわくだき, おんがえし, じしん
1884	リングマ, じしん, はかいこうせん, おんがえし, ずつき
1865	バンギラス, ずつき, れいとうビーム, じしん, １０まんボルト
1863	サンダー, どろかけ, はかいこうせん, ドリルくちばし, １０まんボルト
1853	ランターン, かげぶんしん, １０まんボルト, ハイドロポンプ, れいとうビーム

▼表 5.4　反復 1 内での対戦トップ 10 プレイヤーのパーティ

レート	パーティ
1869	カビゴン, バブルこうせん, かえんほうしゃ, おんがえし, ロケットずつき
1861	カビゴン, かえんほうしゃ, すてみタックル, かみなり, れいとうパンチ
1858	カビゴン, ばくれつパンチ, かえんほうしゃ, のしかかり, バブルこうせん
1848	カビゴン, かえんほうしゃ, ばくれつパンチ, のしかかり, バブルこうせん
1842	カビゴン, のしかかり, ふぶき, でんじほう, かえんほうしゃ
1829	ライコウ, どろかけ, １０まんボルト, どくどく, いあいぎり
1824	カビゴン, れいとうパンチ, だいもんじ, のしかかり, かみなり
1817	カビゴン, のしかかり, だいもんじ, なみのり, かいりき
1812	カビゴン, なみのり, ソーラービーム, ふぶき, すてみタックル
1811	カビゴン, ふぶき, ころがる, すてみタックル, かみなり

▼表 5.5　反復 2 内での対戦トップ 10 プレイヤーのパーティ

レート	パーティ
1899	カビゴン, のしかかり, なみのり, ロケットずつき, ころがる
1887	カビゴン, じしん, だいもんじ, のしかかり, すてみタックル
1868	カビゴン, ころがる, ソーラービーム, かみなりパンチ, のしかかり
1862	カビゴン, のしかかり, ちきゅうなげ, どろかけ, ソーラービーム
1841	カビゴン, のしかかり, ほのおのパンチ, れいとうビーム, ふぶき
1831	ラプラス, つのドリル, ソーラービーム, ハイドロポンプ, １０まんボルト
1831	カビゴン, ばくれつパンチ, のしかかり, でんじほう, はかいこうせん
1825	カビゴン, なみのり, かいりき, のしかかり, ちきゅうなげ
1819	カビゴン, ソーラービーム, どくどく, かえんほうしゃ, のしかかり
1814	ケンタロス, つのドリル, ふみつけ, じしん, のしかかり



▲図 5.2　反復 (iteration) ごとのプレイヤーの平均レート

▼表 5.6　反復 9 内での対戦トップ 10 プレイヤーのパーティ

レート	パーティ
1875	カビゴン, ほのおのパンチ, おんがえし, ソーラービーム, すなあらし
1866	カビゴン, とっしん, バブルこうせん, すてみタックル, だいもんじ
1838	カビゴン, すてみタックル, でんじほう, だいもんじ, ソーラービーム
1834	カビゴン, ずつき, じしん, ちきゅうなげ, おんがえし
1817	カビゴン, じしん, すなあらし, のしかかり, れいとうパンチ
1814	カビゴン, ばくれつパンチ, のしかかり, れいとうビーム, ころがる
1812	カビゴン, おんがえし, だいもんじ, すなあらし, なみのり
1810	カビゴン, すてみタックル, ころがる, ソーラービーム, どろかけ
1806	カビゴン, ふぶき, のしかかり, どろかけ, じしん
1801	カビゴン, ちきゅうなげ, おんがえし, かげぶんしん, かえんほうしゃ

5.3　全反復でのパーティ混合での対戦結果

反復的な学習によって強くなっていっているのかを確認します。各反復内でのレーティングバトル結果の上位 100 プレイヤーを抽出し、それを 10 反復分混合し、合計 1,000 プレイヤーでレーティングバトルさせます。上位を抽出する理由は、各反復で学習対象としているパーティ群にはランダム生成パーティを混合していることおよび多様性確保のために強さが劣るパーティも混合されているためです。

1,000 プレイヤーのレーティングバトルの結果として得られる各プレイヤーのレートを、それが所属する反復ごとに平均した結果を図 5.2 に示します。

反復 0 はもちろんランダム生成パーティなので弱いのは当然ですが、パーティ・行動双

方が最適化された反復 1 でも飽和せず、反復 2 のほうがより強くなっています。しかし、それ以上は強くなっていないようで、学習手法の限界に達しているようです。改良の余地は残りますが、反復的なパーティ生成と強化学習により、戦略が改善できることがわかりました。

▼表 5.7　全反復混合での上位 10 プレイヤーのパーティ

レート	パーティ
1795	ドンファン, じしん, げんしのちから, ころがる, つのでつく
1776	バンギラス, じしん, かいりき, げんしのちから, ばくれつパンチ
1775	ドンファン, つのでつく, じしん, いわくだき, げんしのちから
1764	ドンファン, ころがる, げんしのちから, じしん, つのでつく
1747	ドンファン, じしん, ころがる, つのでつく, げんしのちから
1744	カビゴン, れいとうパンチ, すなあらし, のしかかり, ばくれつパンチ
1742	ドンファン, じしん, ころがる, かいりき, げんしのちから
1732	カビゴン, どろかけ, おんがえし, バブルこうせん, ばくれつパンチ
1729	ニドクイン, じしん, つのドリル, はかいこうせん, だいもんじ
1722	カビゴン, じしん, のしかかり, かみなり, ソーラービーム

レーティングバトル上位 10 プレイヤーを表 5.7 に示します。驚くべきことに、最上位はカビゴンではなくドンファンという結果になりました。ドンファンは 1,000 パーティ中 7 パーティでしか登場していないのですが、そのうち 5 パーティが上位 10 パーティに入るという極めて良い結果を出しています。「5.5 なぜドンファンが最上位だったか」で定性的に考察します。なお、下位 10 プレイヤーは表 5.8 に示すものになりました。いずれも反復 0 で生成されたプレイヤーで、技構成だけでなくバトル中の行動もよくなかったものと考えられます。

▼表 5.8　全反復混合での下位 10 プレイヤーのパーティ

レート	パーティ
1054	ニョロトノ, はかいこうせん, なみのり, サイコキネシス, いわくだき
1080	カメックス, ばくれつパンチ, ハイドロポンプ, でんじほう, たきのぼり
1114	スターミー, おんがえし, たきのぼり, かみなり, すてみタックル
1115	ポリゴン２, かみなり, でんじほう, １０まんボルト, ふぶき
1119	スイクン, バブルこうせん, いわくだき, たきのぼり, おんがえし
1125	ゴルダック, ハイドロポンプ, スピードスター, はなびらのまい, れいとうパンチ
1129	ヤドキング, サイコキネシス, じしん, とっしん, のしかかり
1132	オーダイル, おんがえし, かいりき, ハイドロポンプ, げんしのちから
1143	カイリュー, かみなり, れいとうパンチ, どろかけ, どくどく
1148	エレブー, いわくだき, でんじほう, どくどく, すてみタックル

5.4　バトル中の行動の分析

上位のプレイヤーが、バトル中にどんな行動をとっていたのか定性的に確認します。

ドンファン, じしん, げんしのちから, ころがる, つのでつく

1 位のプレイヤーが用いたパーティです。基本的にはじしんを連打し、飛行タイプのサンダーにはげんしのちから（岩タイプ）を使い分けていました。他の技はほぼ使われません。ラプラス・ゴルダックなど水タイプの相手と対面すると、なみのり等でなすすべなく負けてしまいます。ずつき・おんがえしを連打してくるケンタロスに対しても攻め勝てています。

カビゴン, れいとうパンチ, すなあらし, のしかかり, ばくれつパンチ

6 位のプレイヤーが用いたパーティです。基本的にのしかかりを使い、のしかかりが半減となるバンギラスにはばくれつパンチ、カイリューには 4 倍弱点を突けるれいとうパンチを使っていました。

バンギラス, げんしのちから, １０まんボルト, ばくれつパンチ, かえんほうしゃ

11 位のプレイヤーが用いたパーティです。4 タイプの攻撃技を覚えています。げんしのちからはサンダー・ラプラスに、１０まんボルトはゴルダック・オーダイル・ニョロト

ノ（いずれも水単タイプで、入力特徴量上は区別がつかない）に、ばくれつパンチ（格闘タイプ）はカビゴン・バンギラスに、かえんほうしゃはライコウに使うという形でうまく使い分けができていました。なお、ライコウには攻め負けます。

ケンタロス, つのドリル, のしかかり, どくどく, つのでつく

12 位のプレイヤーが用いたパーティです。ほぼすべてのバトルでつのドリル（命中率 30% の一撃必殺技）を連打するという恐ろしい戦法でした。この環境では、強力な格闘技を受けたり、相手がゴーストタイプでつのドリルが効かないという状況がほとんどありませんでした。ほとんどのバトルで相手に倒される前に 3 回以上つのドリルを試行する機会があり、試行回数 2 回で 51%、3 回で 66% の確率で相手を倒せるという計算になります。1vs1 バトルなので交代でゴーストタイプが出てくることを一切考慮しなくて済むため、極めて単純ながら強力な戦法といえます。

過去の学習ではタイプ相性的に「こうかはいまひとつ」の技を選んでしまうことも多々あり難がありましたが、今回は攻撃技の選択に関して明らかな間違いはほとんどなく、十分な学習結果となっていました。相手のポケモンによって選ぶ技はほぼ固定されており、状況に応じた選択はなされていませんでしたが、攻撃技を使うだけであれば適切な戦略の範疇です。どくどく等の補助技はほとんど使えていない状況は依然として続きます。1vs1 バトルでは交代の隙というのがないので、能力アップよりひたすら攻撃することの有効性が高いとも考えられます。

5.5 なぜドンファンが最上位だったか

ドンファンは各反復内でのレーティングバトルでは上位 10 パーティに入っていなかったにもかかわらず、全反復混合のレーティングバトルでは 1 位になりました。この理由を考察します。

全反復混合のレーティングバトルでの、1,000 パーティ中のポケモンの出現回数上位 10 種類を表 5.9 に示します。

▼表 5.9　全反復混合レーティングバトルに参加した全パーティ中のポケモンの出現回数

要素	出現回数
カビゴン	468
ケンタロス	92
ライコウ	75
サンダー	69
ラプラス	61
バンギラス	51
ガルーラ	28
カイリュー	19
リングマ	16
エーフィ	12

全体の 40% 以上をカビゴンが占めるという状況になっています。この環境では、カビゴンに勝てるか否かが決定的な差になります。

ドンファンの主力技じしんと、カビゴンの主力技のしかかりでダメージ計算を行い、相手を何発で倒せるかを表 5.10 に示します。

▼表 5.10　ドンファンとカビゴンが相手を倒すのに必要なターン数

攻撃側＼防御側	ドンファン	カビゴン
ドンファン (じしん)	3〜4 発	3〜4 発
カビゴン (のしかかり)	4〜5 発	4 発

素早さはドンファンのほうが速いことを加味すると、急所・のしかかりによる麻痺が生じない限りドンファンが勝ちということがわかります。仮にカビゴンがドンファンに対して効果抜群となるなみのりを使用すると 3 発、れいとうパンチを使用すると 3〜4 発です。これらの技を選択すれば、確定ではありませんがドンファンに勝てる可能性はあります。ただ依然として乱数で勝敗が決まるので、単純にのしかかりを連打して麻痺に賭けるほうが単純で汎用性が高そうです。この環境においては、ドンファンはカビゴンを上回っているといえそうです。なお、人間同士のポケモンバトルの公式大会のルールでドンファンが日の目を見たことはないと思われますが、それは同じタイプのガラガラのほうが持ち物「ふといホネ」によって攻撃力が高くなるためです。持ち物なしのルールになっているために絶妙な攻撃力と防御力のバランスで注目対象になったといえます。逆に各反復内のレーティングバトルではドンファンが上位に来なかった理由は、水タイプの相手に対して勝ち目がなく、水タイプのポケモンが出てくる確率が相対的に高い環境で順位が下がったのだと考えられます。結論として、ドンファンが最上位に来た理由は、カビゴンの出現頻

度が極めて高い、かつ天敵となる水タイプが相対的に少ない環境となっていたこと、その中でカビゴンに攻め勝てる性能を持っていたためだといえるでしょう。同時に、環境内のポケモンの出現頻度によってどのポケモンが統計的に強いかが変わるというのは根本的な難題だといえそうです。

ところで、人間同士の対戦ではカビゴンに対してどう対処するのが正解なのでしょうか。ポケモンバトルに関する非公式の考察本から引用します。

> (カビゴンについて) 第二世代の任天堂公式ルール『ニンテンドウカップ 2000』における絶対王者と呼べるポケモン。
> (対策の節) まず、カビゴン対策の筆頭候補となるポケモンがムウマである。(中略) これはムウマがカビゴン対策の役割しか担当しない場合の話である。ムウマはカビゴンを殴り倒せるポケモンではなく、「ほろびのうた」や「みちづれ」の効果によって共倒れにすることで、カビゴン対策の仕事をこなすことができる。
> ―ポケットモンスター赤緑青ピカチュウ＆金銀クリスタル マップ＆ずかん＆対戦考察本『ポケモンバトルノスタルジアＤＸ』上巻 (発行:つうしんケーブルクラブ) より

ということで、共倒れ前提の戦法が薦められています。つまり、今回採用している 1vs1 ルールではそもそも実行不可能です。今回は相手のポケモンごとに決まった 1 つの技を連打するという戦略が有効で、あまり賢い AI にはなりませんでした。その中で、ドンファンがカビゴンに対抗できるということは筆者は予想しておらず、興味深い結果が得られたとも考えられます。カビゴンでとにかく攻撃技を打ちまくるという単純な戦略に対して有効な賢い手段を学習するには、3vs3 ルールへと移行することが必須のようです。

5.6 第 3 巻でカビゴンが登場しなかった理由

第 3 巻でもポケモン金銀環境でパーティ生成、強化学習を行っていたにもかかわらず、実験結果中の強かったパーティにカビゴンが含まれていませんでした。この理由について考察します。

第 3 巻でのパーティ生成手法は本巻と似ていますが、パーティ評価関数が違いました。まずランダムに生成したパーティをランダムに行動を選択するエージェントで戦わせてレートを計算し、次にパーティを入力、レートを出力とする関数を教師あり学習していました。第 3 巻ではすべてのポケモン・技を含めた 3vs3 バトルというルールでしたが、本巻のルール上で改めてランダムに生成したパーティをランダムに行動を選択するエージェントで戦わせました。129 種類のポケモンが各 10 回ずつパーティに含まれるようにパーティを 1,290 個生成しました。

▼表 5.11　本巻のルールでランダムに行動した結果のレート上位 5 プレイヤーのパーティ

レート	パーティ
1887	カビゴン, のしかかり, じしん, だいもんじ, はかいこうせん
1864	スターミー, なみのり, ハイドロポンプ, サイコキネシス, でんじほう
1854	ゲンガー, ギガドレイン, でんじほう, れいとうパンチ, かみなりパンチ
1852	バンギラス, ばくれつパンチ, げんしのちから, じしん, おんがえし
1827	ゲンガー, １０まんボルト, ちきゅうなげ, おんがえし, かげぶんしん

▼表 5.12　ポケモンごとの平均レート上位 10 件

レート	パーティ
1758	ゲンガー
1726	バンギラス
1690	ハッサム
1671	カイリュー
1660	ハガネール
1659	ムウマ
1657	ケンタロス
1652	オーダイル
1649	エアームド
1640	レアコイル

対戦結果のレートが上位のパーティを表 5.11 に示します。パーティごとのレートでは、カビゴンが 1 位という結果でした。各ポケモンは 10 パーティに含まれるので、ポケモンごとにそれを含むパーティのレートの平均を計算した結果を表 5.12 に示します。カビゴンは 18 位で、最上位にはタイプ相性上の耐性が優秀なポケモンが来ていることが見て取れます。タイプ相性を考えずにランダムに技を選ぶことが要因であると考えられます。

さらに、パーティ生成の条件として「1.2 本書で扱うポケモンバトルのルール設定について」で選択した 52 個の技という条件を撤廃し、ほぼすべての技をパーティに含めた場合で同様の実験を行いました。その結果、カビゴンを含むパーティの平均レートは 129 ポケモン中 50 位まで後退しました。その理由は、カビゴンが「じばく」を覚えることにありました。ランダムに使えば負けに直結する技なので、これを覚えたカビゴンのレートが非常に低く、統計的にカビゴンの評価を下げる要因となったようです。ランダムに行動した結果の勝敗を用いると、本来の強さとかけ離れた結果になってしまうことを示しています。本巻で提案したような、強化学習の結果をパーティ生成へフィードバックする仕組みが重要であるといえます。

第 6 章

まとめ

本巻では、あらゆるパーティのバトル中の行動を選択できる単一のモデル「汎用行動選択モデル」を中心に据え、その学習手段と応用について提案しました。従来は、学習のしやすさからパーティごとに別々のモデルパラメータを一から学習していました。しかし多種多様なパーティを生成し、それらすべてを適切に行動させた上で強いパーティと戦法を発見するには効率が悪いという問題がありました。汎用行動選択モデルでは入力に自分のパーティ情報を加えることで特定のパーティと学習されるパラメータを分離し、異なるパーティ間でも戦法の共有を可能にしました。第 2 章では、汎用行動選択モデルのパラメータ数の適切な規模を決定するため、過去に動作実績があるパーティごとのモデルが選択した行動を疑似的な教師データとして教師あり学習を行い、様々なパーティの行動を 1 つのモデルで選択できることを確認しました。第 3 章では、本来の目的となる強化学習により教師あり学習でのモデルより強いモデルの学習を試みました。その過程で Optuna を用いたハイパーパラメータチューニングを行い、学習率が特に重要なハイパーパラメータであることが判明しました。第 4 章では、バトル中の行動を選択するためのモデルが出力する Q 値の最初のターンでの値がパーティの強さや相手との相性の良さを反映していることを発見し、この値を最大化することにより強いパーティを生成する手法を提案しました。第 5 章では、パーティ生成とバトル中の行動の強化学習を交互に反復することによって、双方の最適化を図りました。人間同士の対戦で非常に強力なポケモンとされたカビゴンが上位に現れ、またバトル中の技の選択結果を観察したところタイプ相性を適切に考慮した内容となっていました。1vs1 バトルという限定的な環境ではありますが、単一のシンプルなモデルでポケモンバトルのフェーズ全体をカバーすることができ、また定性的にも妥当な結果が得られました。今後は、行動として交代が絡んでくる 3vs3 バトルへの対応を中心として本巻で提案した手法の応用範囲を広げていくことが課題となります。

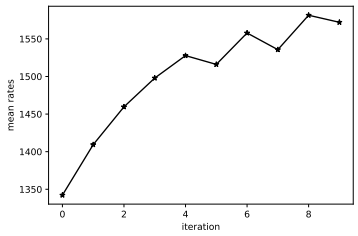
付録 A

カビゴン禁止ルール

本編の実験結果では、カビゴンが王者として君臨し、いかにそれを倒すかという形で他のポケモンがランク付けされるという形になりました。この付録では、カビゴンの使用を禁止する特別ルールを設けた場合の実験結果を示します。パーティ生成での使用可能ポケモンからカビゴンを除外するという 1 点だけを変更し、その他の条件はすべて第 5 章と同様としました。

各反復で生成された上位プレイヤー 100 個ずつを混合した 1,000 プレイヤーでのレーティングバトルの結果を示します。図 A.1 は各プレイヤーのレートを、それが所属する反復ごとに平均した結果です。本編では反復 2 で強さが飽和していましたが、今回の条件では強さが飽和していません。カビゴンを除外したことで圧倒的に強い戦略が簡単には定まらず、試行錯誤が長くかかった可能性があります。パーティに多く出現したポケモンを表 A.1 に示します。本編ではカビゴン 468 回、次点でケンタロス 92 回でしたが、カビゴンの枠をライコウ・サンダーで分け合う形になっています。上位 10 プレイヤーを表 A.2 に示します。カビゴンを除外したにもかかわらず、本編と同様にドンファンがトップになりました。ライコウ・サンダーにそれぞれ有効なじしん・げんしのちからを覚えたドンファンはこの環境でもやはり多数派のポケモンに対して強いという結果です。ニドクインのつのドリル戦法も同じです。カビゴンが抜けた穴をバンギラスが埋めています。カビゴンがよく覚えていた、4 倍弱点のばくれつパンチが飛んでくる頻度が下がったためかもしれません。

結論として、カビゴンが抜けると次点のポケモンが順当に穴を埋める結果となりました。今後 3vs3 ルールへ拡張すればカビゴンを対策するためだけのポケモンというのがパーティに組み込めるかもしれません。その際にはまた違った結果が得られるのではないかと思います。



▲図 A.1　反復 (iteration) ごとのプレイヤーの平均レート

▼表 A.1　全反復混合レーティングバトルに参加した全パーティ中のポケモンの出現回数

要素	出現回数
ライコウ	250
サンダー	198
ケンタロス	108
ラプラス	75
バンギラス	51
サンダース	45
ハピナス	33
ガルーラ	31
カイリュー	31
フーディン	21

▼表 A.2　全反復混合での上位 10 プレイヤーのパーティ

レート	パーティ
1937	ドンファン, ころがる, じしん, つのでつく, げんしのちから
1901	ニドクイン, つのドリル, れいとうパンチ, すなあらし, じしん
1900	バンギラス, げんしのちから, じしん, ばくれつパンチ, １０まんボルト
1822	ニドクイン, れいとうパンチ, つのドリル, じしん, ほのおのパンチ
1822	ドンファン, つのでつく, いわくだき, じしん, げんしのちから
1820	ライコウ, かみなり, おんがえし, すなあらし, スピードスター
1818	ドンファン, ずつき, どくどく, じしん, げんしのちから
1807	バンギラス, いあいぎり, げんしのちから, じしん, だいもんじ
1800	バンギラス, かえんほうしゃ, ずつき, じしん, げんしのちから
1799	バンギラス, げんしのちから, じしん, １０まんボルト, なみのり

あとがき

ポケモン金銀のルールで研究し始めて 2 冊目になります。前巻では金銀でできるようになった新たな要素を取り込んでとりあえずバトルができるようにする、ということを目指しました。見た目は派手になったのですが学習結果はいまいち、という印象で、アルゴリズムにせよハイパーパラメータ調整にせよ足元を固める必要があるとの思いで研究を進めたのが今回のテーマになります。バトルの内容としては比較的地味な内容にはなってしまいましたが、着実に技術が進歩していると感じています。ポケモン金銀といえばカビゴンが最強というのはよく知られた事実ですが、今回のルール制約故とはいえ意外な対抗馬が出てきたのは面白かったです。これからも筆者を楽しませてくれる面白い戦法が出てくることに期待しています。

趣味でこうした研究に時間を割いていると、ゲームを手動で遊ぶ時間がなかなか取れず、気づいたらポケモン第 8 世代（ソード・シールド）が終わってしまうんじゃないかと不安になってきています。早くいいアルゴリズムを開発し、パソコンで 1 か月ぐらい動かし続けたら勝手にすごい戦法が出てくるような状況にして、遊ぶ時間を確保したいところです。

同人誌の発行という観点では、新型コロナウイルスの影響によりオフラインでのイベントが開催されず非常に寂しい限りです。来年こそはイベントが復活してくれることを願っています。

ここまでお付き合いいただきありがとうございました。また第 5 巻でお会いしましょう。

ＰｏｋéＡＩ　＃４：金銀汎用行動選択モデル編

2020 年 9 月 12 日　技術書典 9 v1.4.0

著　者　select766 (select766@outlook.jp)
発行所　ヤマブキ計算所
印刷所　（電子書籍配布専用）

ヤマブキ計算所

#4.5: 汎用行動選択モデル編 DLC

このパートは、技術書典 9（2020 年 9 月）発行の第 4 巻の執筆から技術書典 13（2022 年 9 月）までの間の開発の進捗をまとめたものです。実質的には 2021 年 4 月までの研究です。コロナ禍が明けた際に第 4 巻の第 2 版として物理本を出版したいと考えておりましたが、結局 2022 年も万全な状態でのイベント開催が難しい見通しとなりました。これ以上発表を遅らせないため、電子媒体での発表としました。本編では 1vs1 のバトルを扱っていましたが、この DLC（Download Contents）では 3vs3 のバトルへの拡張を解説します。

目次

第 1 章

汎用行動選択モデルの 3vs3 への拡張

1.1　イントロダクション

本編では 1vs1 ルールを実装しましたが、本章では 3vs3 ルールへの拡張を解説します。3vs3 ルールはパーティにポケモンを 3 匹入れるルールで、1vs1 の場合と違い行動として技を選ぶ以外に、控えポケモンとの交代が追加されます。最初に、交代を考慮できるようにモデルの構造を変更します。次に、行動の選択肢の増加や決着がつくまでのターン数の増加などにより強化学習が難しくなることに対処するテクニックを追加します。最後に、パーティ生成機構にも調整を加えたうえで強化学習機構と組み合わせ、パーティ生成と行動選択の交互学習を実現します。

1.2　入力特徴量の変更

汎用行動選択モデルにおける行動選択の仕組みは、DNN で表現された Q 関数に選択肢を表現したベクトルを入力し、その出力が最大となるような選択肢を選ぶというものでした。1vs1 バトルでの選択肢は場に出ているポケモンが覚えている技 4 つのうちどれを選択するかでした。3vs3 バトルでは、技のほかに、控えのポケモン（最大 2 匹）のいずれかと交代する選択肢が追加されます。選択肢を表現するベクトルが交代を表現できるようにすれば、3vs3 バトルが可能になると考えられます。1vs1 バトルでの選択肢ベクトルは、251 種類の技それぞれに対応する次元を持つ 251 次元のベクトルでした。また、状態ベクトルに場に出ている自分のポケモンの種族を表す 251 次元のベクトルが含まれていました。3vs3 バトルでは、ポケモンの種族に対応するベクトルを状態ベクトルから選択肢ベクトルに移し、さらに交代、強制交代に対応する各 1 次元の要素を加えて表現することと

しました。強制交代とは、場に出ているポケモンが瀕死となった際に交代先となる控えのポケモンを選択する場合を指します。3vs3 バトルのために追加した要素は交代・強制交代の 2 次元だけです。3vs3 バトルのために拡張した選択肢ベクトルを表 1.1 に示します。

▼表 1.1　3vs3 バトルにおける選択肢ベクトル

特徴	次元数	説明
交代	1	選択肢が交代（強制交代も含む）を表すとき 1
強制交代	1	選択肢が強制交代を表すとき 1
ポケモン	251	技の選択肢では、場に出ているポケモンに対応する次元を 1 にする。 交代・強制交代の選択肢では、交代して繰り出すポケモンに対応する次元を 1 にする。
技	267	技の選択肢では、繰り出す技に対応する次元を 1 にする。 交代・強制交代の選択肢では、交代して繰り出すポケモンが 覚えている技すべてに対応する次元を 1 にする。

この入力で交代を含めた行動の選択が可能であると考えた理由は、1vs1 バトルでの Q 関数の出力を観察した結果です。本編第 4.1 節での観察から、場に出ている自分のポケモンと相手のポケモンの相性にあわせて Q 値が変動していることがわかります。相性の情報は、状態特徴量における相手のタイプと、選択肢特徴量における自分のポケモンの種族から判定するように Q 関数が学習されたと考えられます。仮にその Q 関数の入力として自分の場に出ているポケモンを表す部分に交代先のポケモンの情報を入れた場合、そのポケモンが相手のポケモンに対して相性が良いのであれば高い Q 値、相性が悪ければ低い Q 値が得られ、自然に交代を行動選択に考慮可能であると考えました。技の部分についても、相手のポケモンに有効な技を持っていれば高い Q 値が得られると期待できます。ただし、交代に 1 ターンかかるので場に出ているポケモンが技を出す場合と比べ、同じ技であっても Q 値を割り引く必要があります。それを考慮可能とするため、選択肢特徴量に交代であるかどうかの情報を含めました。

細かいこととして、ポケモン金銀における技は 251 種類ですが、めざめるパワーについて、タイプ不定のもののほか、16 タイプそれぞれを別の技として追加してあります。ただ当面はめざめるパワーを覚えたポケモンは使用しません。実装上は持ち物に対応する次元も存在していますが、本書の範囲では持ち物は持たせていません。パーティに含められるポケモン・技については、本編第 1.2 節で選定したものに限定しています。

1.2.1　実験

まずは 1vs1 と同様の学習パラメータで強化学習を行っていきます。

DNN モデルは 16 チャンネル 3 層です。バトル 10 万回で学習しました。

バトル 10 万回で学習した最終的なモデルと、途中の 1 万回時点のチェックポイントと、ランダムに行動するエージェントがそれぞれ学習時と同じ 1,000 パーティを操作し、合計 3,000 プレイヤーでレーティングバトルを行いました。各エージェントが操作したプレイヤーの平均レートを表 1.2 に示します。

▼表 1.2　学習時のバトル回数による強さの比較

エージェント	平均レート
1 万回時点	1530
10 万回時点	**1589**
ランダム行動	1380

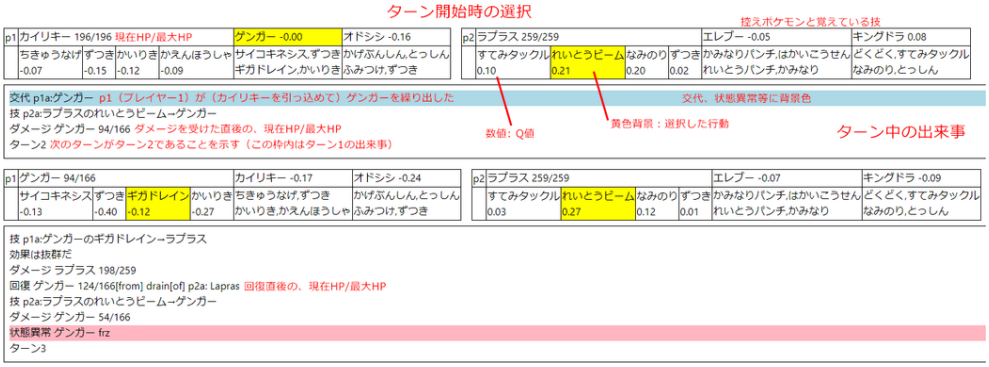
強化学習により、ランダムより強いエージェントが作れること、また学習が進むことで強さが向上することがわかりました。

1vs1 バトルでは DNN の構造は 16 チャンネル 3 層がパラメータチューニングの結果最良でしたが、より複雑な戦略が要求される 3vs3 バトルではよりパラメータ数が多いモデルが良い可能性があります。そこで、64 チャンネル 3 層のモデルを学習しました。他の条件は変更していません。同様にレーティングバトルを行った結果を表 1.3 示します。

▼表 1.3　チャンネル数を増加させた場合の強さの比較

エージェント	平均レート
64 チャンネル 3 層	**1584**
16 チャンネル 3 層	1560
ランダム行動	1355

若干ですが 64 チャンネル 3 層のモデルのほうが強いことがわかりました。次節以降、モデルの構造はこれを採用します。



▲図 1.1　可視化結果の読み方

1.2.2　行動の可視化

強化学習により、ランダムに行動するより良いモデルが学習できました。本項では、モデルを用いてバトルした場合の実際の行動を可視化します。64 チャンネル 3 層のモデルを使い、レーティングバトルで最も強かったパーティについて観察しました。パーティの内容は以下の通りです。

```
ラプラス,LV55,すてみタックル,れいとうビーム,なみのり,ずつき
エレブー,LV50,かみなりパンチ,はかいこうせん,れいとうパンチ,かみなり
キングドラ,LV50,どくどく,すてみタックル,なみのり,とっしん
```

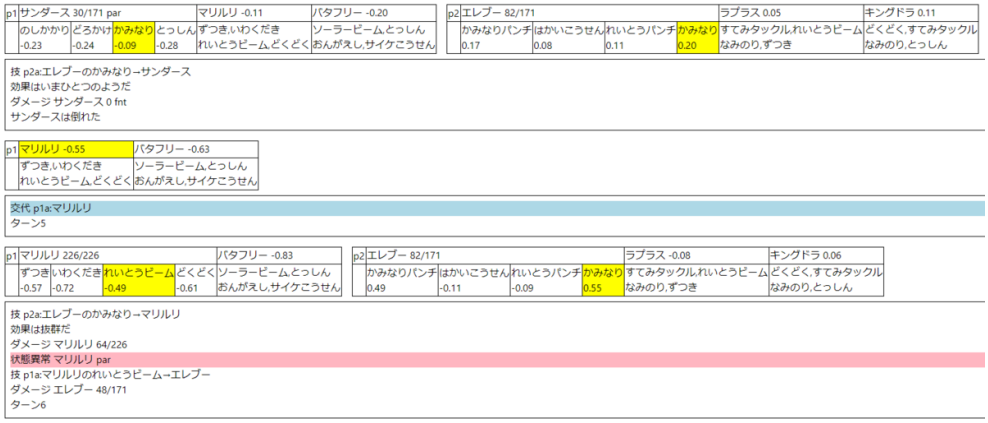
可視化結果の読み方を図 1.1 に示します。

図 1.2 では、エレブーの相手がサンダースの時はどの技でも Q 値は似ていますが、相手がマリルリになると、効果が抜群な電気タイプの技の Q 値が高くなっています。タイプ相性を把握していると考えられます。

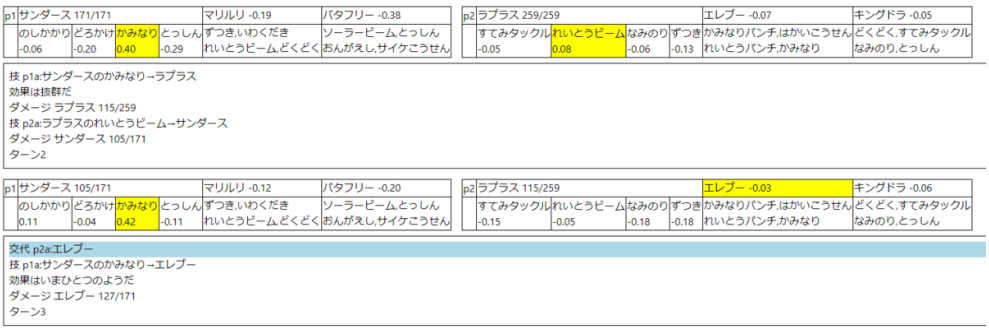
図 1.3 では、ラプラスの相手がサンダースで、相性が悪いです。しかし最初のターンでは交代せず、ダメージを受けた次のターンで交代しています。交代をうまく使えているとは言いにくいです。

図 1.4 では、ラプラスの相手がレアコイルで、相性が悪いので交代しようとしています。キングドラに交代し 1 ターン攻撃した後、さらにエレブーに交代するという挙動をしており、戦略が迷走しています。

このパーティが強いのは、ラプラスのれいとうビームの通りが良いからというのが主な



▲図 1.2　可視化結果 1。観察対象パーティは p2 （右）側。



▲図 1.3　可視化結果 2。観察対象パーティは p2 （右）側。

理由だと思われます。3vs3 バトルでは交代という行動が増えた点がポイントですが、それを有効に活用できているとはいえない結果でした。

1.3　補助報酬

前節でバトルログを可視化してわかったのは、学習したエージェントの行動でタイプ相性などがある程度考慮されているものの、まだまだ改善の余地が大きいということでした。本節から、強化学習手法の改善を試みていきます。

▲図 1.4　可視化結果 3。観察対象パーティは p1 （左）側。

今回は、初代ルールの 3vs3[*1]のときに提案した、補助報酬を与える手法を金銀ルールでも試してみます。通常の強化学習では勝敗に対してのみ報酬が得られますが、3vs3 では 1vs1 と比べて勝敗が決するまでのターン数（行動数）が多くなるため、各ターンの行動の良し悪しを判断しづらくなります。より短い期間で行動を評価できるよう、相手にダメージを与えたり、相手を 1 体倒した時点で少量の報酬を与えることを考えます。

勝敗以外に強化学習アルゴリズムに与える報酬を補助報酬と呼ぶことにし、以下のように定式化します。

- ターン t における状態を以下の記号で表す。（自分のポケモンが倒れた場合にも行動選択が生じるので、厳密にはターンではなく t 回目の行動）
  - $H_f(t)$ = 自分の各ポケモンの現在 HP/残り HP の 3 体の平均
  - $H_e(t)$ = 相手の各ポケモンの現在 HP/残り HP の 3 体の平均
  - $A_f(t)$ = 瀕死でない自分のポケモンの数 / 3 体
  - $A_e(t)$ = 瀕死でない相手のポケモンの数 / 3 体
- ターン t における自分の有利さを $P(t) = \lambda_H(H_f(t) - H_e(t)) + \lambda_A(A_f(t) - A_e(t))$ とする
  - λ_H, λ_A は正の定数（ハイパーパラメータ）
- ターン t における行動に対する補助報酬を、$P(t+1) - P(t)$ とする

$H_f(t)$ はバトル開始時に 1、自分が全滅すれば 0 になります。ほかの式も同様です。HP

[*1] https://select766.hatenablog.com/entry/2019/01/21/223807 書籍では第 2 巻に掲載

が 1 でも残っている状態と完全に倒してしまうのは状況に差があるため、HP とは別に瀕死でないポケモン数の指標を含めています。$P(t)$ は、自分が全滅に近ければ小さな値、相手が全滅に近ければ大きな値をとることになり、自分が有利な状態であるほど大きな値になるといえます。そして、ターン間の差をとることにより自分の有利が拡大すれば正の報酬、不利になれば負の報酬が与えられることになります。

この手法を実装し、実験しました。$\lambda_H = \lambda_A = 0.25$ とし、他の学習条件は前節と同じです。補助報酬を含めて学習したモデルと含めずに学習したモデル、ランダムに行動するモデルの 3 つでレーティングバトルを行い、平均レートを比較しました。

▼表 1.4　補助報酬の有無によるモデルの強さの比較

モデル	平均レート
ランダム	1340
補助報酬無し	1567
補助報酬あり	**1593**

実験結果を表 1.4 に示します。補助報酬を加えることにより、定量的に改善することがわかりました。ただバトル中の行動を観察すると、HP が減ったポケモンを控えのポケモンと交代する無駄な行動がみられました。自分のポケモンが倒されると負の報酬が発生するため、これを回避するため、勝敗全体で見れば不利な行動をとってしまう場合があるようです。短期的な報酬を与えることによる負の側面であるといえます。

さらに、ハイパーパラメータ調整として λ_H, λ_A を変化させてみましたが、元のパラメータ $\lambda_H = \lambda_A = 0.25$ が最善でした。さらに、補助報酬のアルゴリズムを変更したものを 2 つ試しました。1 つ目は、相手に関する値だけを計算に含めるものです。すなわち、$H_f(t) = A_f(t) = 0$ とします。自分のポケモンの HP が減っているときに、倒されて負の報酬を受けるのを回避するために無駄に交代するという挙動を避ける目的です。しかし、変更しないのが良いことがわかりました。2 つ目のアルゴリズムは、ゲーム終了時に今までの補助報酬を打ち消す補助報酬を与えるものです。すなわち、バトル終了時の報酬として、勝敗によるものに過去ターンの補助報酬の総和の符号を反転させたものを加算します。目的は、僅差で勝利した場合も圧勝した場合もバトル全体での合計の報酬を同じにすることにより、被害の大きさにかかわらず勝敗に注力させることです。しかしながらこの手法も、結果にほとんど影響がないようでした。

1.4 バトル数の調整

モデルの強化学習におけるバトル数は、1vs1 の時のパラメータを引き継いで 10 万に設定していました。3vs3 ではそもそもバトル 1 回あたりのターン数が違うこと、学習すべき行動がより複雑であることから、この値が適切なのかどうか検証しました。

バトル数を 10 万より長くするにあたり、エージェントがランダムに行動する確率 ϵ の減衰後の下限 ϵ_{min} を新たにパラメータとして加えました。ステップ数 $step$ に対するランダム行動率 $\epsilon(step)$ は、減衰率 ϵ_{decay} を用いて $\epsilon(step) = \max\{(1-\epsilon_{decay})^{step}\epsilon, \epsilon_{min}\}$ と定義します。今回、$\epsilon = 0.3, \epsilon_{decay} = 2 \times 10^{-6}, \epsilon_{min} = 0.01$ に設定しました。この場合、およそ 50 万ステップで $\epsilon(step)$ が下限になります。なおバトル 1 回あたりの 1 エージェントの行動回数は平均 20 回程度で、バトルに参加する 2 プレイヤー両方の行動を学習サンプルとして用いるので、40 ステップ分になります。すなわち 1.25 万バトルで下限に達するということです。

1 万バトルごとにモデルを保存するようにして、50 万バトルまで学習を進めてみました。学習率は従来通り 1×10^{-5} です。バトル数 1 万、3 万、10 万、30 万、50 万の時点のモデルをレーティングバトルで比較しました。ランダムは弱すぎて、比較対象間の細かい差を測ることに悪影響があるため外しています。

▼表 1.5　学習バトル数を変化させた場合の強さの比較

学習バトル数	平均レート
10000	1449
30000	1496
100000	1510
300000	**1528**
500000	1517

結果を表 1.5 に示します。1 万から 10 万にかけては大きな改善が見られる一方、それ以上は伸びが弱く、30 万から 50 万にかけてはむしろ弱くなっています。実験に、10 万バトルで 24 時間程度かかります。現状のアルゴリズムのままバトル数を伸ばすのはコストのわりにメリットが少ないようなので、10 万バトルを今後の実験でも標準のパラメータとして使います。

1.5 学習率の調整

強化学習はアルゴリズムが正しくてもハイパーパラメータ設定により性能が大幅に変わってきます。1vs1 バトル環境において、Optuna による調整を行ったところ学習率が最も大きな要素であることがわかりました（本編第 3.5.1 項）。しかしこのパラメータのまま 3vs3 バトル環境へ適用すると学習が不安定となったため、学習率を下げていました。3vs3 に最適な学習率を調べました。最適な学習率はモデルのレイヤー数などに大きく左右されるため、具体的な数字に一般性はありません。

異なる学習率で 1 万バトルおよび 10 万バトル強化学習させ、そのエージェント同士をレーティングバトルさせて平均レートを測定しました。学習・評価用パーティはランダムに生成した 1,000 パーティです。

▼表 1.6　学習率を変化させた場合の強さの比較（1 万バトル）

学習率	平均レート
3×10^{-6}	1479
1×10^{-5}	1512
3×10^{-5}	1512
1×10^{-4}	**1516**
3×10^{-4}	1482

▼表 1.7　学習率を変化させた場合の強さの比較（10 万バトル）

学習率	平均レート
3×10^{-6}	**1515**
1×10^{-5}	1514
3×10^{-5}	1503
1×10^{-4}	1481
3×10^{-4}	1487

1 万バトル時点（表 1.6）では、学習率 $1 \times 10^{-4} \sim 1 \times 10^{-5}$ が良い領域で、小さくても大きくても強さが低下します。10 万バトル（表 1.7）では最良の学習率が 3×10^{-6} でしたが、1×10^{-5} とはわずかな差でした。結論としては、今回の実験設定では 1×10^{-5} 程度を設定するのが安定しそうです。しいて言えば、学習の最初のほうは学習率高めで、徐々に下げていくのが望ましいと考えられます。教師あり学習では徐々に学習率を下げるというテクニックは当たり前に行われています。学習率を下げるタイミングには、学習を

進めても正解率が変化しなくなったときという目印が使えます。しかし強化学習ではこのような使いやすい指標がないので、応用は比較的難しいといえます。

1.6 強化学習中の対戦相手の選択

ポケモンバトルの強化学習の特徴的な要素として、バトル開始時にエージェントに操作すべきパーティが割り当てられるという点があります。バトルごとに異なるパーティが割り当てられ、相手のパーティとの関係性により有利不利がある程度決まっています。つまり、パーティの組み合わせによって、いくら最善の行動を選択したとしても勝てない相手と当たったり、また戦力差が大きすぎてランダムに行動していても勝ってしまったりする場合が考えられます。強化学習では適切な行動を選択した場合に高い報酬が得られ、そうでない場合に低い報酬が得られるという枠組みによってエージェントがより適切な行動を選択できるように学習していきます。そのため、どんな行動を選んでも結果が変わらない、また相手にかかわらず同じ行動を選び続けても結果が変わらないという状況は強化学習に悪影響を及ぼすと考えられます。事前に結果を述べると、本節で提案する手法はあまりうまくいきませんでした。

現在の学習システムでは、学習対象となる 1,000 パーティから毎回ランダムに 2 パーティを選択し、エージェントに操作させて対戦させます。パーティ選択の結果、次のような組み合わせになる場合もあり得ます。

```
パーティ1:
  ラプラス,LV55,すてみタックル,れいとうビーム,なみのり,ずつき
  エレブー,LV50,かみなりパンチ,はかいこうせん,れいとうパンチ,かみなり
  キングドラ,LV50,どくどく,すてみタックル,なみのり,とっしん

パーティ2:
  バタフリー,LV55,かげぶんしん,どくのこな,はかいこうせん,ギガドレイン
  ギャラドス,LV50,すなあらし,ロケットずつき,かげぶんしん,ふぶき
  アリアドス,LV50,ギガドレイン,ナイトヘッド,サイケこうせん,サイコキネシス
```

このような組み合わせだと、ポケモン自体の強さや技構成に差がありすぎて、パーティ 1 はれいとうビームを連打するだけで勝ち、パーティ 2 は何をしても負けという状況になってしまいます。より強化学習の効果を高めるためには、強さが均衡しているパーティ同士を対戦させ、行動によって勝敗に影響が生じるべきであると考えられます。このアイデアを実現するため、学習中の勝敗に基づきパーティのレートを計算し、そのレートが近いもの同士を対戦させる手法を提案します。より具体的には、次のようなアルゴリズムになります。

1. 全パーティにレート 1500 を割り当てる。
2. 各パーティのレート + 乱数（正規分布（平均 0, 標準偏差 200））をソートし、隣接するパーティ同士を 1 回ずつ対戦させる。
3. 対戦結果に応じ、各パーティのレートを変動させ、2. に戻る。勝ったパーティのレートが上昇、負けたパーティのレートが下降する。

なお、強さは同程度だがタイプ相性上一方的な展開になるような組み合わせについては回避できません。また、学習中はエージェントの行動にランダム探索（ε-greedy）が入っているため、緻密な行動が必要となるパーティのレートが低く見積もられることに注意が必要です。

学習中にパーティに割り当てられたレートの例（最大最小 5 パーティずつ）を表示します。

```
レート=1976
スイクン,LV55,バブルこうせん,どろかけ,おんがえし,たきのぼり
ライコウ,LV50,いわくだき,１０まんボルト,どろかけ,スピードスター
ブースター,LV50,すてみタックル,どくどく,のしかかり,でんじほう

レート=1907
フリーザー,LV50,おんがえし,すてみタックル,れいとうビーム,かげぶんしん
レアコイル,LV50,どくどく,スピードスター,でんじほう,かみなり
サンダース,LV55,かげぶんしん,ずつき,どろかけ,かみなり

レート=1899
カビゴン,LV55,のしかかり,じしん,どろかけ,いわくだき
ギャラドス,LV50,れいとうビーム,バブルこうせん,ふぶき,なみのり
ファイヤー,LV50,すてみタックル,どろかけ,はがねのつばさ,ゴッドバード

レート=1889
サンダース,LV55,１０まんボルト,ロケットずつき,のしかかり,おんがえし
オドシシ,LV50,サイコキネシス,ずつき,スピードスター,どろかけ
プテラ,LV50,どくどく,じしん,そらをとぶ,げんしのちから

レート=1881
ラプラス,LV55,すてみタックル,れいとうビーム,なみのり,ずつき
エレブー,LV50,かみなりパンチ,はかいこうせん,れいとうパンチ,かみなり
キングドラ,LV50,どくどく,すてみタックル,なみのり,とっしん
```

```
レート=941
バタフリー,LV55,かげぶんしん,どくのこな,はかいこうせん,ギガドレイン
ギャラドス,LV50,すなあらし,ロケットずつき,かげぶんしん,ふぶき
アリアドス,LV50,ギガドレイン,ナイトヘッド,サイケこうせん,サイコキネシス

レート=1037
ウソッキー,LV50,かみなりパンチ,ほのおのパンチ,かげぶんしん,ばくれつパンチ
```

```
ハピナス,LV55,どくどく,ソーラービーム,すなあらし,バブルこうせん
ダグトリオ,LV50,げんしのちから,どくどく,いわくだき,どろかけ

レート=1074
ヨルノズク,LV50,はかいこうせん,スピードスター,どろかけ,つばさでうつ
カモネギ,LV55,ロケットずつき,スピードスター,かげぶんしん,どくどく
フシギバナ,LV50,すてみタックル,かげぶんしん,ずつき,ロケットずつき

レート=1074
パラセクト,LV55,どくどく,どくのこな,ロケットずつき,とっしん
キレイハナ,LV50,どくのこな,はかいこうせん,どくどく,ソーラービーム
エレブー,LV50,はかいこうせん,おんがえし,のしかかり,サイコキネシス

レート=1082
フシギバナ,LV50,かげぶんしん,どろかけ,いあいぎり,とっしん
ヨルノズク,LV55,はがねのつばさ,スピードスター,どくどく,かげぶんしん
ラッタ,LV50,かみなり,ふぶき,すてみタックル,いわくだき
```

定性的に見て、妥当な序列になっているように見えます。提案手法で学習したモデルを従来手法とレーティングバトルで比較しました。学習に用いたパーティ群と同じパーティ群で対戦しています。

▼表 1.8　対戦相手を選択する手法による強さの比較

手法	平均レート
提案	1504
従来	1496

結果を表 1.8 に示します。残念ながら、わずかな差しかありませんでした。別の問題として、Q 値の大きさでパーティの強さを判定するというパーティ生成手法との相性が悪いことがわかりました。バトル開始時点では勝つか負けるか五分五分という状況ばかりを学習するため、パーティ自体の強さが Q 値に反映されません。実際には、最強クラスのパーティは自分より弱い相手との対戦のほうが多くなるため若干は強さが Q 値に現れますが、実際のデータで確認したところ対戦相手をランダムに選択する場合と比べ、パーティ間での Q 値の分散は小さめでした。そのため、本節の手法は次節以降で使用していません。

1.7　パーティ生成手法の調整

1vs1 バトルの時と同様に、強化学習の結果得られる、行動ごとの価値の期待値を学習した Q 関数を用いて強力なパーティを生成することを試みます。手法は 1vs1 バトルの時のものを応用し、3vs3 対応のための変更を行います。

3vs3 対応の変更点は、(1) 隣接パーティの生成部分と、(2) 類似したパーティの生成を避けるためのペナルティ部分です。(1) については、10% の確率で、パーティ内のポケモ

ン 1 匹を別のポケモンに変更し、そのポケモンの技はランダムに初期化します。なお、レベル配分は LV55,50,50 となっており、変更前のレベルを維持します。最初のパーティ生成時に、どのポケモンが LV55 になるかはランダムです。残りの 90% は、ポケモン 1 匹を選択してその技 1 つをランダムに変更します。(2) については、パーティ間の類似度測定の関数において、1vs1 のときはパーティを表す特徴量=P,M,PM,MM を用いていました。3vs3 ではポケモンのペアの類似度が加わり、P,M,PM,MM,PP の 5 種類の特徴量を用いた内積になります。パーティが完全一致したときの類似度は、3+12+12+18+3=48 となり、1vs1 の時の最大値 15 より大きくなります。そのため、類似したパーティが生成された場合のペナルティの強さ λ を調整し、$\lambda = 0.1$ を用いることとしました。

パーティ生成の実験を行いました。強化学習で得られたモデルの Q 関数を用いて生成されたパーティをランダムに 10 個抽出して示します。

```
ゲンガー,LV50,ギガドレイン,ばくれつパンチ,かみなり,ほのおのパンチ
デンリュウ,LV55,かいりき,ばくれつパンチ,ほのおのパンチ,かみなり
ラッタ,LV50,かみなり,バブルこうせん,かいりき,ふぶき

リングマ,LV50,おんがえし,どろかけ,スピードスター,どくどく
カモネギ,LV55,スピードスター,おんがえし,どろかけ,どくどく
スイクン,LV50,どくどく,おんがえし,スピードスター,どろかけ

リングマ,LV50,ずつき,れいとうパンチ,おんがえし,ほのおのパンチ
ゴルダック,LV50,おんがえし,ずつき,ハイドロポンプ,ロケットずつき
キングドラ,LV55,ずつき,ハイドロポンプ,おんがえし,ロケットずつき

リングマ,LV50,おんがえし,かみなりパンチ,ちきゅうなげ,ずつき
ハピナス,LV50,ずつき,おんがえし,ソーラービーム,ちきゅうなげ
スリーパー,LV55,かみなりパンチ,ちきゅうなげ,おんがえし,ずつき

ルージュラ,LV55,どくどく,サイコキネシス,れいとうビーム,おんがえし
ナッシー,LV50,サイコキネシス,やどりぎのタネ,どくどく,おんがえし
ムウマ,LV50,おんがえし,サイケこうせん,サイコキネシス,どくどく

ルージュラ,LV50,サイコキネシス,とっしん,れいとうビーム,どろかけ
エアームド,LV50,いあいぎり,スピードスター,おんがえし,ドリルくちばし
カブトプス,LV55,ちきゅうなげ,とっしん,れいとうビーム,いあいぎり

ルージュラ,LV55,れいとうビーム,ロケットずつき,ずつき,サイコキネシス
ハピナス,LV50,ふぶき,ロケットずつき,れいとうビーム,ずつき
ニドクイン,LV50,ずつき,ロケットずつき,ふぶき,れいとうビーム

リングマ,LV55,スピードスター,おんがえし,いあいぎり,ころがる
フォレトス,LV50,ころがる,おんがえし,ギガドレイン,スピードスター
エアームド,LV50,スピードスター,いあいぎり,おんがえし,ゴッドバード

エンテイ,LV50,かげぶんしん,かえんほうしゃ,どくどく,はかいこうせん
マルマイン,LV55,１０まんボルト,かげぶんしん,はかいこうせん,どくどく
パルシェン,LV50,はかいこうせん,どくどく,かげぶんしん,ふぶき
```

```
リングマ,LV50,ころがる,おんがえし,ちきゅうなげ,ずつき
ベロリンガ,LV50,れいとうビーム,ちきゅうなげ,ころがる,ずつき
サイドン,LV55,ころがる,ちきゅうなげ,ずつき,れいとうビーム
```

ある程度強いパーティが生成できていると考えられます。カビゴンやサンダーが含まれていないのは、ランダムに生成したパーティ 1,000 個で学習しているため、それらのポケモンの強さを認識できていないものと考えられます。

1.8 パーティ生成と行動選択の交互学習

3vs3 バトルでの強化学習とパーティ生成の手法・実装が完成したので、1vs1 バトルの時と同様、これらを交互に動作させて強いパーティとその適切な運用法を学習します。

1.8.1 実験条件

アルゴリズム自体は 1vs1 の時（本編第 5 章）と変わりません。

- 反復回数 10
- パーティ数
  - Q 関数を用いて生成するパーティ数 871
  - ランダム生成するパーティ数 129
- パーティの生成条件
  - ポケモンの制限 最終進化系 129 種類（ミュウツー・ドーブル等除く）
  - 技の制限 効果が見込める 52 種類（高威力攻撃技主体）
- 類似パーティ生成抑制のペナルティ 0.1
- 強化学習
  - 探索: ε -greedy
  - ランダム行動する確率 ε : 0.3
  - ε decay: 2.0×10^{-6}
  - 最小の ε : 0.01
  - 報酬割引率: 0.95
  - バッチサイズ: 32
  - 最初に学習するまでのステップ数（replay buffer のサンプル数）: 500
  - N サンプル収集するたびに optimize: 1
  - N optimize ごとに target network のアップデート: 100
  - replay buffer サイズ: 100,000

- optimizer （Adam）の学習率: 0.00001
- バトル数: 100,000
- 補助報酬
  - 瀕死でないポケモン数 0.25
  - 残り HP 率 0.25

1.8.2 各反復の結果

各反復ではパーティが 1,000 個生成されます。そして、そのパーティを用いて学習したモデルを用いてレーティングバトルを行い、上位 10 パーティを示します。また、上位 100 パーティに含まれるポケモン・技の分布（それぞれ出現回数トップ 10）を示します。全 1,000 パーティでの分布では、多様性を確保したパーティ生成の影響で、ほとんどのポケモンが覚える技（かげぶんしん、どくどく等）が常に上位を占めて傾向が出ませんでした。

反復 0

▼表 1.9　反復 0 で生成されたパーティの上位 10 個

レート	パーティ
1975	カビゴン,LV50, かみなりパンチ, のしかかり, すてみタックル, ふぶき シャワーズ,LV50, のしかかり, ふぶき, バブルこうせん, ずつき エレブー,LV55, おんがえし, サイコキネシス, スピードスター, はかいこうせん
1969	バクフーン,LV50, すてみタックル, かえんほうしゃ, じしん, ほのおのパンチ オドシシ,LV50, ずつき, どろかけ, じしん, おんがえし カビゴン,LV55, のしかかり, はかいこうせん, ちきゅうなげ, いわくだき
1941	ケンタロス,LV50, でんじほう, つのドリル, とっしん, おんがえし ゴルダック,LV50, かいりき, のしかかり, れいとうビーム, なみのり ルージュラ,LV55, れいとうビーム, おんがえし, ロケットずつき, とっしん
1939	バンギラス,LV55, じしん, どろかけ, いわくだき, げんしのちから ムウマ,LV50, １０まんボルト, かみなり, スピードスター, でんじほう シャワーズ,LV50, どくどく, はかいこうせん, かげぶんしん, なみのり
1930	カイリュー,LV55, おんがえし, ロケットずつき, れいとうビーム, だいもんじ シャワーズ,LV50, たきのぼり, どろかけ, とっしん, ロケットずつき ゲンガー,LV50, のしかかり, ナイトヘッド, サイコキネシス, ほのおのパンチ
1899	ジュゴン,LV50, ロケットずつき, ずつき, つのドリル, れいとうビーム カビゴン,LV55, すなあらし, ソーラービーム, のしかかり, でんじほう ハピナス,LV50, かえんほうしゃ, すてみタックル, とっしん, どくどく
1881	ハピナス,LV50, バブルこうせん, １０まんボルト, だいもんじ, かえんほうしゃ シャワーズ,LV50, れいとうビーム, とっしん, ロケットずつき, すてみタックル バクフーン,LV55, スピードスター, ばくれつパンチ, かみなりパンチ, だいもんじ
1878	ドンファン,LV50, すなあらし, すてみタックル, ころがる, じしん ランターン,LV50, でんじほう, なみのり, ハイドロポンプ, おんがえし スイクン,LV55, バブルこうせん, れいとうビーム, どくどく, はかいこうせん
1870	シャワーズ,LV55, ふぶき, とっしん, スピードスター, なみのり フーディン,LV50, ちきゅうなげ, サイコキネシス, ほのおのパンチ, れいとうパンチ マンタイン,LV50, おんがえし, ふぶき, どろかけ, なみのり
1845	プテラ,LV50, いわくだき, とっしん, そらをとぶ, おんがえし ラプラス,LV55, ふぶき, いわくだき, なみのり, ずつき ライコウ,LV50, すなあらし, かみなり, でんじほう, はかいこうせん

反復 0（すなわちパーティはランダム生成）で生成されたパーティを表 1.9 に示します。強化学習により行動を学習したうえでパーティをランキングすると、強いポケモン・技の傾向が見て取れます。

▼表 1.10　反復 0 の上位 100 パーティにおけるポケモンの出現回数

要素	出現回数
シャワーズ	11
カビゴン	10
ランターン	9
ライコウ	9
サンダー	9
ケンタロス	7
ムウマ	7
ゲンガー	7
ハピナス	6
ヌオー	6

▼表 1.11　反復 0 の上位 100 パーティにおける技の出現回数

要素	出現回数
おんがえし	83
のしかかり	58
どくどく	58
はかいこうせん	56
かげぶんしん	54
ずつき	53
とっしん	53
スピードスター	43
どろかけ	41
すてみタックル	40

上位 100 パーティにおけるポケモン・技の出現回数の集計を表 1.10、表 1.11 に示します。パーティはランダム生成ですが、上位 100 パーティを抽出すればカビゴン、サンダーなどの出現回数が大きくなりました。技については、エージェントが現状使いこなせていないものの覚えるポケモンが多いどくどく、かげぶんしんが上位に来ています。

反復 1

▼表 1.12　反復 1 で生成されたパーティの上位 10 個

レート	パーティ
1922	カビゴン,LV55, のしかかり, ずつき, じしん, かみなり サンダース,LV50, スピードスター, かみなり, ずつき, のしかかり ムウマ,LV50, サイケこうせん, ずつき, かみなり, スピードスター
1903	ファイヤー,LV50, どろかけ, はがねのつばさ, スピードスター, だいもんじ ライコウ,LV55, いあいぎり, １０まんボルト, どろかけ, かいりき サイドン,LV50, じしん, ふみつけ, だいもんじ, １０まんボルト
1877	カビゴン,LV55, かえんほうしゃ, サイコキネシス, のしかかり, じしん サンダー,LV50, ドリルくちばし, １０まんボルト, どろかけ, かみなり ドククラゲ,LV50, ふぶき, なみのり, ハイドロポンプ, ヘドロばくだん
1874	ガルーラ,LV55, だいもんじ, のしかかり, じしん, いわくだき ケンタロス,LV50, いわくだき, のしかかり, じしん, だいもんじ プテラ,LV50, ゴッドバード, だいもんじ, じしん, いわくだき
1866	カビゴン,LV55, じしん, かみなりパンチ, どろかけ, のしかかり ミルタンク,LV50, かみなりパンチ, のしかかり, どろかけ, じしん エアームド,LV50, どろかけ, ゴッドバード, はがねのつばさ, そらをとぶ
1854	プテラ,LV50, じしん, げんしのちから, はかいこうせん, ゴッドバード フシギバナ,LV50, はっぱカッター, げんしのちから, はかいこうせん, やどりぎのタネ ライコウ,LV55, はかいこうせん, １０まんボルト, でんじほう, かみなり
1852	カビゴン,LV55, のしかかり, バブルこうせん, でんじほう, ロケットずつき ペルシアン,LV50, バブルこうせん, でんじほう, ロケットずつき, のしかかり フシギバナ,LV50, やどりぎのタネ, ロケットずつき, どくのこな, はっぱカッター
1839	サンダー,LV50, とっしん, ドリルくちばし, おんがえし, １０まんボルト ケンタロス,LV55, とっしん, おんがえし, １０まんボルト, れいとうビーム ランターン,LV50, １０まんボルト, れいとうビーム, おんがえし, とっしん
1835	カイリュー,LV55, １０まんボルト, かえんほうしゃ, ハイドロポンプ, のしかかり サイドン,LV50, １０まんボルト, かえんほうしゃ, おんがえし, のしかかり カビゴン,LV50, かえんほうしゃ, おんがえし, １０まんボルト, のしかかり
1823	カビゴン,LV55, のしかかり, サイコキネシス, だいもんじ, ばくれつパンチ スリーパー,LV50, ばくれつパンチ, サイコキネシス, れいとうパンチ, でんじほう ベトベトン,LV50, れいとうパンチ, だいもんじ, ばくれつパンチ, でんじほう

反復 1 は、反復 0 で強化学習された Q 関数を用いて強いパーティを生成し、さらにそのパーティの運用を強化学習した結果になります。強化学習後のエージェントでレーティングバトルを行った上位 10 パーティを表 1.12 に示します。カビゴン、サンダーの優位性がよく現れる状況になりました。

▼表 1.13　反復 1 の上位 100 パーティにおけるポケモンの出現回数

要素	出現回数
カビゴン	37
サンダー	21
ライコウ	15
バンギラス	15
プテラ	8
サンダース	6
ガルーラ	6
ベトベトン	6
ヌオー	6
キングドラ	6

▼表 1.14　反復 1 の上位 100 パーティにおける技の出現回数

要素	出現回数
のしかかり	92
１０まんボルト	85
おんがえし	64
じしん	63
ずつき	49
でんじほう	46
どろかけ	42
はかいこうせん	38
とっしん	37
どくどく	37

ポケモン・技の集計を表 1.13、表 1.14 に示します。カビゴン、サンダーが 1 位、2 位となり、その主力技となるのしかかり、１０まんボルトが技の上位に来ました。

反復 2

▼表 1.15　反復 2 で生成されたパーティの上位 10 個

レート	パーティ
	カビゴン,LV55, ふぶき, なみのり, かみなり, のしかかり
	ヤドキング,LV50, なみのり, ふぶき, バブルこうせん, どくどく
1926	スターミー,LV50, ふぶき, どくどく, バブルこうせん, かみなり
	カビゴン,LV55, かいりき, のしかかり, じしん, ソーラービーム
	ミルタンク,LV50, のしかかり, どろかけ, じしん, かいりき
1900	メガニウム,LV50, のしかかり, ソーラービーム, かいりき, どろかけ
	カビゴン,LV55, はかいこうせん, どくどく, れいとうビーム, おんがえし
	メガニウム,LV50, どくどく, はかいこうせん, おんがえし, やどりぎのタネ
1886	パルシェン,LV50, おんがえし, はかいこうせん, どくどく, れいとうビーム
	カビゴン,LV50, なみのり, のしかかり, じしん, ロケットずつき
	ヌオー,LV50, はかいこうせん, のしかかり, なみのり, じしん
1885	ファイヤー,LV55, そらをとぶ, はがねのつばさ, はかいこうせん, だいもんじ
	カビゴン,LV55, のしかかり, じしん, だいもんじ, はかいこうせん
	エンテイ,LV50, はかいこうせん, ふみつけ, だいもんじ, おんがえし
1883	キュウコン,LV50, だいもんじ, のしかかり, おんがえし, はかいこうせん
	カビゴン,LV55, じしん, ちきゅうなげ, のしかかり, どろかけ
	ニョロボン,LV50, ちきゅうなげ, どろかけ, ハイドロポンプ, のしかかり
1870	サワムラー,LV50, かいりき, ロケットずつき, どろかけ, のしかかり
	カビゴン,LV50, ばくれつパンチ, すてみタックル, のしかかり, れいとうビーム
	カメックス,LV50, れいとうビーム, どくどく, すてみタックル, ばくれつパンチ
1868	ガルーラ,LV55, どくどく, のしかかり, れいとうビーム, すてみタックル
	カビゴン,LV55, じしん, はかいこうせん, れいとうビーム, おんがえし
	イノムー,LV50, はかいこうせん, じしん, おんがえし, れいとうビーム
1867	ブラッキー,LV50, ロケットずつき, おんがえし, いあいぎり, はかいこうせん
	カビゴン,LV50, れいとうビーム, かいりき, じしん, のしかかり
	サンドパン,LV50, のしかかり, どくどく, かいりき, じしん
1858	ジュゴン,LV55, どくどく, かいりき, のしかかり, れいとうビーム
	カビゴン,LV50, のしかかり, いわくだき, れいとうビーム, ロケットずつき
	スイクン,LV55, たきのぼり, おんがえし, いあいぎり, ふぶき
1858	ハッサム,LV50, おんがえし, ロケットずつき, スピードスター, いあいぎり

反復 2 での上位パーティを表 1.15 に示します。カビゴン無双になりました。特に気になるのは、パーティの先頭にカビゴンが集中している点です。次項で考察します。

▼表 1.16　反復 2 の上位 100 パーティにおけるポケモンの出現回数

要素	出現回数
カビゴン	56
サンダー	15
ガルーラ	10
カメックス	9
スターミー	8
メガニウム	6
ヌオー	6
ファイヤー	6
イノムー	6
バンギラス	6

▼表 1.17　反復 2 の上位 100 パーティにおける技の出現回数

要素	出現回数
のしかかり	104
じしん	93
れいとうビーム	62
おんがえし	57
すてみタックル	51
はかいこうせん	44
かいりき	42
とっしん	35
どろかけ	33
ばくれつパンチ	33

ポケモン・技の集計を表 1.16、表 1.17 に示します。出現頻度がカビゴンに強く偏っています。

反復 9 （最終）

▼表 1.18　反復 9 で生成されたパーティの上位 10 個

レート	パーティ
1906	カビゴン,LV50, ほのおのパンチ, でんじほう, じしん, おんがえし ミルタンク,LV55, おんがえし, じしん, でんじほう, ほのおのパンチ デンリュウ,LV50, でんじほう, ばくれつパンチ, おんがえし, ほのおのパンチ
1883	カビゴン,LV55, のしかかり, バブルこうせん, でんじほう, ちきゅうなげ マンタイン,LV50, スピードスター, れいとうビーム, たきのぼり, ハイドロポンプ カメックス,LV50, ハイドロポンプ, たきのぼり, れいとうビーム, ちきゅうなげ
1876	カビゴン,LV55, ソーラービーム, だいもんじ, れいとうパンチ, のしかかり キングラー,LV50, ふみつけ, のしかかり, どくどく, どろかけ ゴローニャ,LV50, だいもんじ, どくどく, どろかけ, のしかかり
1861	カビゴン,LV50, ほのおのパンチ, なみのり, どくどく, のしかかり ヤドキング,LV55, どくどく, ふみつけ, なみのり, のしかかり ケンタロス,LV50, ふみつけ, なみのり, どくどく, のしかかり
1859	カビゴン,LV55, すなあらし, じしん, ちきゅうなげ, のしかかり ドンファン,LV50, じしん, つのでつく, のしかかり, すなあらし ヘラクロス,LV50, つのでつく, いあいぎり, ちきゅうなげ, じしん
1857	カビゴン,LV55, だいもんじ, はかいこうせん, のしかかり, じしん ガルーラ,LV50, はかいこうせん, じしん, どくどく, だいもんじ マグカルゴ,LV50, だいもんじ, はかいこうせん, どくどく, じしん
1856	スイクン,LV50, なみのり, れいとうビーム, おんがえし, かげぶんしん ドンファン,LV55, つのでつく, かげぶんしん, じしん, おんがえし ケンタロス,LV50, じしん, つのでつく, かげぶんしん, おんがえし
1831	カビゴン,LV50, ばくれつパンチ, ふぶき, かえんほうしゃ, のしかかり プクリン,LV55, のしかかり, かえんほうしゃ, ふぶき, ばくれつパンチ ニョロボン,LV50, ばくれつパンチ, いわくだき, ふぶき, のしかかり
1830	カビゴン,LV50, ロケットずつき, のしかかり, かみなりパンチ, じしん ブースター,LV50, ロケットずつき, どろかけ, のしかかり, でんじほう スリーパー,LV55, ロケットずつき, のしかかり, でんじほう, かみなりパンチ
1829	カビゴン,LV55, ばくれつパンチ, バブルこうせん, かみなりパンチ, のしかかり ニョロボン,LV50, のしかかり, バブルこうせん, ばくれつパンチ, れいとうビーム パルシェン,LV50, れいとうビーム, バブルこうせん, スピードスター, ふぶき

最終反復である反復 9 で得られた上位のパーティを表 1.18 に示します。対戦結果ではカビゴンがやはり無双している状況でした。1vs1 バトルでカビゴンへの対策として現れたドンファンが、3vs3 バトルではカビゴンと同じパーティに組み込まれ、上位に食い込んでいます。

▼表 1.19　反復 9 の上位 100 パーティにおけるポケモンの出現回数

要素	出現回数
カビゴン	49
ライコウ	14
サンダー	13
バンギラス	13
ドンファン	8
ガルーラ	7
バクフーン	6
プテラ	6
カメックス	5
サイドン	5

▼表 1.20　反復 9 の上位 100 パーティにおける技の出現回数

要素	出現回数
のしかかり	93
じしん	91
おんがえし	62
どろかけ	53
１０まんボルト	47
どくどく	40
れいとうビーム	38
はかいこうせん	38
いわくだき	37
すてみタックル	37

ポケモン・技の集計を表 1.19、表 1.20 に示します。カビゴンが圧倒的上位のまま収束したという状況になりました。

1.8.3　パーティの先頭にカビゴンが多い理由

パーティ生成の中でカビゴンが強力なポケモンであることは十分認識されていることがわかりましたが、パーティ先頭にカビゴンが置かれていることが非常に多い傾向がみられます。本項ではその理由について考察します。反復 9 のトップ 100 パーティにおいて、出現回数上位 5 ポケモンについてそれがパーティ先頭に現れた回数、先頭以外に現れた回数を抽出し、表 1.21 に示します。

▼表 1.21　上位パーティ内における、パーティ先頭およびそれ以外でのポケモンの出現回数の集計

要素	パーティ先頭での出現回数	パーティ先頭以外での出現回数
カビゴン	48	1
バンギラス	13	0
サンダー	11	2
ライコウ	10	4
スターミー	5	0

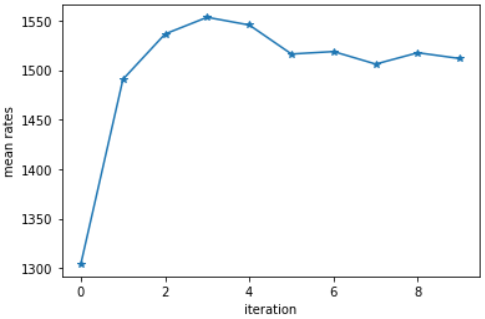
主要なポケモンは圧倒的にパーティ先頭に集中しています。なお、パーティ先頭以外だけで出現回数の順位を確認すると、ドンファンの 8 回が最大となりポケモン間での偏りが小さいことがわかりました。この結果は、カビゴン・バンギラス・サンダーを並べたパーティが生成されていないということでもあり最良のパーティ構築ができていない可能性を示唆しています。パーティ生成のアルゴリズムは、バトル開始時の Q 関数の出力の全行動に対する最大値を最大とするようなパーティを生成するというものでした。Q 関数の入力として自分のパーティが反映される部分を展開すると、Q(先頭ポケモン, 技 1)、Q(先頭ポケモン, 技 2)、…、Q(控えポケモン 1 に交代, その技すべて)、Q(控えポケモン 2 に交代, その技すべて) という 6 つの値のうちの最大値、すなわち最善手をとったときの勝率を取ることになります。バトル開始時にいきなり交代が最善手になる状況は不利であり、先頭に天敵の少ない強力なポケモンが配置されていればほとんど起こらないと考えられます。この場合、控えポケモンは何であっても最大値に影響しないことになります。Q 関数を用いた手法は、控えポケモンの反映が弱くなるという問題を抱えていることがわかりました。この問題の解決は将来の課題です。

1.8.4　全反復のパーティを混合した評価

ここまで、パーティ生成と学習の反復により、各反復で生成されるパーティが定性的に改善されていることを確認しました。次に、各反復での学習結果で上位 100 パーティを抽出し、反復 10 回分のパーティを混合した 1,000 パーティでレーティングバトルさせます。反復間での強さの違いを確認するため、パーティが抽出された反復ごとにレートの平均を計算し、図 1.5 に示します。

反復 3 までは強さが向上していますが、そのあとは若干弱くなってしまっています。現状の学習手法では、計算時間を延ばしても強さの向上に限界があることがわかりました。

レートトップ 10 のパーティを表 1.22 に示します。やはりカビゴンが極めて強いという結果になりました。前項で考察したように、控えのポケモンについては課題が残ります。



▲図 1.5　各反復で強かったパーティを混合して対戦させてレートを付与し、反復ごとにレートの平均を算出した結果

最後に、レートがトップだったパーティの行動を定性的に確認しました。

```
カビゴン,LV55,かいりき,のしかかり,じしん,ソーラービーム
ミルタンク,LV50,のしかかり,どろかけ,じしん,かいりき
メガニウム,LV50,のしかかり,ソーラービーム,かいりき,どろかけ
```

- カビゴン（自分）VS カビゴン（相手）
  - のしかかりを使う。LV55 なので、相手が LV50 の場合は押し切れる。
- カビゴン VS バンギラス
  - じしんを使う。

基本的に、カビゴンが圧倒的に強いためごり押しになります。タイプ相性によってのしかかりとじしんを使い分ける単純な戦法でした。1vs1 ではドンファンがカビゴンに有利でしたが、ドンファンはメガニウムのソーラービームで倒されてしまうので補完ができていました。

カビゴン・ゴローニャ・モルフォンのパーティとの対戦では、相手に対し有利なポケモンへの交換を繰り返しつつダメージを蓄積するサイクル戦のようにも見える戦略が見えました。

- カビゴン VS ゴローニャ
  - メガニウムに交代。じしんは使わない。
- メガニウム VS ゴローニャ
  - ソーラービームを使う。

  - 相手のパーティにモルフォンがいる場合、ソーラービームの準備ターンにモルフォンに交代されて受けられる。
- メガニウム VS モルフォン
  - ミルタンクに交代。
  - モルフォンはすてみタックルを使う。
- ミルタンク VS モルフォン
  - のしかかりを使う。
  - 相手はゴローニャに交代。
- ミルタンク VS ゴローニャ
  - じしんを使う。
  - 相手はちきゅうなげを使う。

結局のところ、補助技を使ったトリッキーな戦略というのは現れず、カビゴンの攻撃技で大暴れするのが最適解という結果でした。交代により不利な相手への対策も少しはできるようになったといえます。しかし、交代を繰り返すばかりで無駄にダメージを蓄積してしまうループに陥る場合もあり、交代という選択肢がないほうが良いように思われる状況もありました。

▼表 1.22　全反復混合のレーティングバトルにおける上位 10 パーティ

レート	パーティ
1821	カビゴン,LV55, かいりき, のしかかり, じしん, ソーラービーム ミルタンク,LV50, のしかかり, どろかけ, じしん, かいりき メガニウム,LV50, のしかかり, ソーラービーム, かいりき, どろかけ
1797	カビゴン,LV55, のしかかり, じしん, ソーラービーム, かえんほうしゃ ギャロップ,LV50, つのドリル, ふみつけ, かえんほうしゃ, のしかかり ケンタロス,LV50, ふみつけ, かえんほうしゃ, のしかかり, つのドリル
1789	カビゴン,LV55, のしかかり, じしん, かいりき, いわくだき プテラ,LV50, じしん, スピードスター, いわくだき, はがねのつばさ グライガー,LV50, じしん, かいりき, スピードスター, いわくだき
1784	カビゴン,LV55, じしん, ちきゅうなげ, のしかかり, どろかけ ニョロボン,LV50, ちきゅうなげ, どろかけ, ハイドロポンプ, のしかかり サワムラー,LV50, かいりき, ロケットずつき, どろかけ, のしかかり
1769	カビゴン,LV55, かえんほうしゃ, サイコキネシス, のしかかり, じしん サンダー,LV50, ドリルくちばし, １０まんボルト, どろかけ, かみなり ドククラゲ,LV50, ふぶき, なみのり, ハイドロポンプ, ヘドロばくだん
1768	カビゴン,LV50, じしん, のしかかり, かみなり, すてみタックル ドードリオ,LV55, そらをとぶ, すてみタックル, のしかかり, ゴッドバード ガルーラ,LV50, かみなり, じしん, のしかかり, すてみタックル
1767	カビゴン,LV55, じしん, かみなりパンチ, どろかけ, のしかかり ミルタンク,LV50, かみなりパンチ, のしかかり, どろかけ, じしん エアームド,LV50, どろかけ, ゴッドバード, はがねのつばさ, そらをとぶ
1767	バンギラス,LV55, げんしのちから, １０まんボルト, じしん, かえんほうしゃ ギャラドス,LV50, １０まんボルト, たきのぼり, のしかかり, かえんほうしゃ ガルーラ,LV50, のしかかり, １０まんボルト, じしん, かえんほうしゃ
1765	カビゴン,LV55, おんがえし, サイコキネシス, かえんほうしゃ, ばくれつパンチ ヤドラン,LV50, ばくれつパンチ, かえんほうしゃ, サイコキネシス, おんがえし ムウマ,LV50, おんがえし, サイケこうせん, ずつき, サイコキネシス
1764	カビゴン,LV55, おんがえし, じしん, ロケットずつき, どくどく ハッサム,LV50, おんがえし, どくどく, はがねのつばさ, ロケットずつき スターミー,LV50, どくどく, ロケットずつき, たきのぼり, おんがえし

1.9　まとめ

汎用行動選択モデルを用いた 1vs1 バトルの手法を拡張し、3vs3 バトルへ対応させました。そのために、交代を含めた入出力の拡張、バトルが長くなることに対応した補助報酬の提案などを行いました。これにより、3vs3 バトルでのパーティ生成、行動の強化学習の一連の流れを実現することができました。得られた結果はカビゴンの攻撃技を連打するのが最強で、一部の状況では交代を有効活用するサイクル戦のような戦略を確認することが

できました。補助技の活用はまだ成功しておらず、より長期的な戦略をとれるような学習手法を確立することが今後の課題です。

あとがき

ポケモン金銀のルールで 3vs3 バトルを強化学習できるところまで来ました。行動選択にまだまだ難があり、改良の余地は大いにあると考えています。様々なハイパーパラメータを試すような、独立した学習であれば CPU のコア数に応じて並列化できるのですが、１つのモデルを複数の CPU コアを使って高速に学習させるということはまだできていません。学習が高速になれば、より多くのバトルをこなしたり、複雑な構造のモデルを学習することができるようになり、戦略が改善することが期待できます。そのため、2022 年 9 月時点では高速化の作業に取り組んでいます。別の方向性として、ゲーム木を探索するタイプの手法も検討を始めています。しかしながら乱数がある、相手の情報が隠された状態でのゲーム木を処理するアルゴリズムは複雑で、かなりの勉強を要することになりそうです。

2022 年 11 月にはポケモン第 9 世代（スカーレット・バイオレット）が発売になりますので楽しみです。第 1 巻刊行時（2018 年）はまだ第 7 世代でしたので、ずいぶん時間が経ったように感じます。本の刊行ペースは落ちてしまっていますが、まだまだ研究すべき内容がたくさんあるので今後もお付き合いいただければ幸いです。

ＰｏｋéＡＩ　＃４．５：金銀汎用行動選択モデル編追加ＤＬＣ

2022 年 9 月 10 日　技術書典 13 v1.4.5

著　者　select766 (select766@outlook.jp)
発行所　ヤマブキ計算所
印刷所　（電子書籍配布専用）