修 士 論 文 の 和 文 要 旨

研究科・専攻	大学院　情報理工学研究科　情報・ネットワーク工学専攻　博士前期課程		
氏　　名	久米　洋輝	学籍番号	1731065
論 文 題 目	少ない行動記録から麻雀プレイヤの実力を推定する研究		

要　旨

本論文では、麻雀プレイヤの実力を少ない行動記録から推定した研究について述べる．麻雀プレイヤの実力は平均順位やレート値から推定されるが，おおまかな推定をするにも多くの時間がかかる．例えば±0.1位程度の精度で平均順位を推定しようとすると約500戦必要であり，これは通常250時間以上かかる．

本研究ではより少ない対戦数nで麻雀プレイヤの実力をすることを目指して，次のような2つの実験を行った．

1つ目は，十分に強い人工知能プレイヤ（AIプレイヤ）を用いてエラーレートを計算することでプレイヤの実力を推定する実験である．エラーレートとはバックギャモンで一般的に用いられるプレイヤの実力指標であり，これはAIプレイヤが推定する1行動あたりの価値の減少平均値から見積もられる．エラーレートと平均順位の推定性能を比較した．性能の評価は，人々の実力指標として広く用いられている天鳳の平均レート値との相関係数，決定係数により行った．実験の結果，16戦でのエラーレートと500戦での平均順位は同等かそれ以上の推定性能を持つことが明らかとなった．

2つ目は，ニューラルネットワークを用いてプレイヤの実力推定を試みる実験である．本実験では1つ目の実験よりさらに少ない行動記録からプレイヤの実力を推定できないか試みた．1戦分の行動記録から抽出したいくつかの特徴を用いてプレイヤの実力推定を行った．実力推定に用いる特徴は，行動記録を数値化したものから自己符号化器を用いて抽出した特徴と1戦分の行動記録から得られたエラーレートの2つからなる．推定目標には，その行動記録時点でのプレイヤのレート値を用いた．実験の結果，構築したどのネットワークも目標値のほとんど平均値を出力するような推定をするものと同等であり，プレイヤの実力を推定することはできなかった．

電気通信大学 情報理工学研究科
情報・ネットワーク工学専攻 情報数理工学コース

少ない行動記録から
麻雀プレイヤの実力を推定する研究

平成 31 年 1 月 28 日

学籍番号 1731065

久米洋輝

指導教員 保木邦仁

目 次

1 はじめに

現実社会では，人々の熟達度を測る様々な方法が用いられる．例えば，学生の学力を測るときには試験などが用いられ，野球選手の能力を測るときには打率などが用いられる．このような熟達度の計測は，ときとして計測対象となった人にも利益をもたらし得る．自身の熟達度を知ることにより，上達への目標設定を客観的に行うことが可能となるからである．熟達度の計測はまた，ゲームのエンターテイメント性向上にも貢献し得る．自身の熟達度とかけ離れたプレイヤとプレイすることはエンターテインメント性を低下させるからである．人々の熟達度を測ることは思考力を競うようなゲームにおいても重要である．すでにチェスや囲碁，将棋といった 2 人完全情報ゲームにおいてプレイヤの熟達度を推定する研究がいくつかある．しかし，多人数ゲームや不完全情報ゲームにおいては，熟達度を推定したという報告は少ない．

本研究では，4 人不確定不完全情報ゲームである麻雀に着目する．現状では，麻雀においてプレイヤの実力を測る指標は多くは存在しない．現在，実力を測る指標としてよく用いられるものに平均順位やレート値があるが，ある程度正確に実力を測るには麻雀の性質上，数千回程度の対戦が必要である．

インターネット麻雀サイト「東風荘」の一般的なプレイヤの平均順位は，おおよそ平均 2.45，標準偏差 0.15 の正規分布に従うことが知られている [1]．ここで，実力に応じてプレイヤを上級，中級，初級の 3 つに分けることを考える．一般に明確な指標はないが，上級プレイヤを平均順位 2.4 位未満，中級プレイヤを 2.4 位以上 2.6 位未満，初級プレイヤを 2.6 位以上のプレイヤと考えることとする．この場合，上級プレイヤはプレイヤ全体の上位約 25%，初級プレイヤは下位約 25%である．麻雀はその性質上，このようなおおまかな分類を行うにも多くの対戦数が必要である．大半のプレイヤは $2.1 \leq J \leq 2.9$ の範囲内の実力であり，麻雀の偶然の要素がゲームプレイに強い影響を与えるという性質上，どのプレイヤもおおよそ同じような順位確率 $(\frac{1}{4})$ をとることから，1 戦あたりの順位の分散 A は $A \fallingdotseq 1.25$ と考えることができる．平均順位が J であるプレイヤの順位の標本平均を考える．標本サイズ (対戦数) を N として，標本平均の標準偏差は $\sqrt{\frac{A}{N}}$ と見積もることができる．つまり標本平均の標準偏差は $\sqrt{\frac{1.25}{N}}$ で見積もられる [1]．従って ±0.1 位の範囲で平均順位を推定するためには約 500 戦必要である．500 戦行うには，通常 250 時間以上必要である．そのため，麻雀プレイヤの実力を少ない行動記録から推定したいと考えた．

本研究の目的は，麻雀プレイヤの実力推定を少ない行動記録から行うことである．これを達成するために，以下の 2 つの課題に取り組んだ．1 つ目は，エラーレートからプレイヤの実力を推定する課題である．2 つ目は，麻雀の行動記録からエラーレート以外も特徴抽出を行って実力を推定する課題である．

本論文では，2 章で思考力を競うようなゲームにおける実力推定の先行研究，3 章で本研究に関する基礎知識，4 章で本研究の目的，5 章と 6 章では行った実験について述べる．

2　関連研究

2 章では，本研究に関する思考力を競うようなゲームにおいてプレイヤの実力推定を行っている先行研究について概要を述べる．

2.1　囲碁プレイヤの実力推定

荒木らは，13 路 1 局の棋譜からプレイヤの棋力を推定する手法を提案した [2][3]．この手法は畳み込みニューラルネットワーク (CNN) を用いたものであった．

棋譜には，インターネット碁会所である「囲碁クエスト」のものを用いた．CNN を用いて推定したものはプレイヤのレート値である．N 手目の盤面情報を表すビット列を入力データ，プレイヤのレート値を目標値として CNN を訓練した．盤面情報は，(1) すでに置かれている黒石の位置が 1 でそれ以外が 0 の長さ 13×13 のビット列，(2) すでに置かれている白石の位置が 1 でそれ以外が 0 の長さ 13×13 のビット列，(3) 次の 1 手が 1 でそれ以外が 0 の長さ 13×13 のビット列，の 3 つで表現される．白番もしくは黒番のみに限定して盤面情報を入力するため，1 棋譜あたり $N/2$ の盤面が 3 種類であり，CNN への入力のビット数は $13 \times 13 \times (3N/2)$ である．CNN のネットワーク構成は入力側から，入力層，畳み込み層が 3 つ，全結合層，出力層からなっていた．畳み込み層のフィルタサイズは 3×3 であり，チャネル数はそれぞれ 12，8，8 であった．畳み込み層の活性化関数には正規化線形関数を，学習方法には確率的勾配降下法を用いた．N 手目までの情報を用いる場合，ネットワークの重みの数は $162N + 1,861$ 個である．最終的な達成度として，$N = 50$ の場合では平均二乗誤差の平方根が 250 程度まで減少した．

この先行研究と本研究の類似点は，ニューラルネットワーク (NN) を用いて実力を推定をしている点とゲームの行動記録から実力を推定している点である．畳み込み層を用いている点，ゲーム人工知能 (AI プレイヤ) を用いた実力推定はしていない点，2 人確定完全情報ゲームを題材としている点が異なっている．

2.2　チェスプレイヤの実力推定

Guid らはチェスの AI プレイヤを用いて棋譜からチェスプレイヤの実力を順位付けする手法を提案した [4][5]．また，プレイスタイルを数値化する可能性も示した．

2006 年の報告では，チェスの AI プレイヤ「CRAFTY」の示す評価値を用いて 3 つの指標を計算した．CRAFTY の評価値はポーン何個分勝っているかを推定するものである．1 つ目は CRAFTY の示す最善手の評価値とゲーム記録で行われた行動の評価値の差を 1 行動あたりに平均化したもの (平均損失) である．ただし，評価値が $[-2, 2]$ の範囲外にある場合では計算を行わなかった．2 つ目はある状況における最善手の評価値と次善手の評価値の差 (ゲーム状況の複雑さ) である．3 つ目は，CRAFTY の示す最善手の評価値とゲーム記録で行われた行動の評価値の差が 1 以上であった行動の数である．これらの指標を用いて，チェスプレイヤの実力の順位付けとプレイスタイルの分析を行った．

2011 年の報告では，いくつかのチェスの AI プレイヤの探索の深さを変えながら 2006 年の報告と同様にプレイヤの実力の順位付けをし，結果に差が出るのか調査した．用いた 3 つのチェスの AI プレイヤは人間のトッププレイヤよりも強い AI プレイヤである．探索深さを変化させることによっても AI プレイヤの強さは変化する．1 つの悪手が平均損失に影響しすぎないように AI プレイヤの示す最善手の評価値とゲーム記録で行われた行動の評価値の差の最大値を 3 に設定した．評価値の差が 3 よりも大きいときには差の値を 3.0 として平均損失の計算を行うよ

うに変更した．結果は，AI プレイヤの強さを変化させても推定したプレイヤの実力の順位には差が見られなかった．また，AI プレイヤのプレイスタイルが推定結果に影響がある可能性が確認されたが，チェスチャンピオンの順位付けに大きく影響はしなかった．

AI プレイヤを用いて実力を判断する点，ゲームの行動記録から実力を判断する点が本研究との類似点である．NN やエラーレートを用いていない点，2 人確定完全情報ゲームを題材としている点が異なっている．

2.3 将棋プレイヤの実力推定

山下は，将棋の AI プレイヤを用いて棋譜からプレイヤの実力を推定する手法を提案した [6]．

推定に用いた棋譜は「将棋の棋譜でーたべーす」に 2014 年 6 月 30 日までに登録されていたものを用いた．推定した実力はレート値である．AI プレイヤには「Bonanza 6.0」と「GPSFish」を用いた．AI プレイヤの示す最善手と評価値から，平均悪手，平均好手，AI プレイヤとの一致率などを計算した．将棋の AI プレイヤの評価値は駒の損得を推定するものである．計算時には評価値を 100 で割ったものを用いた．平均悪手は AI プレイヤの示す最善手の評価値とゲーム記録で行われた行動の評価値の差を 1 行動あたりに平均化したものである．ただし平均悪手の計算は，ゲーム記録の行動が AI プレイヤの最善手と異なる行動である場合かつ評価値が最善手よりも下がる場合で行った．また，ゲーム記録の行動が 40 手目以降かつ評価値の絶対値が 10 未満の場合のみを用いた．AI プレイヤの行う探索の深さを変えることで，AI プレイヤの強さを変化させて計算を行った．Bonanza と GPSFish の 2 つの AI プレイヤの評価値を用いて同様の計算を行った．

求めたいくつかの結果から，山下は平均悪手がもっとも実力を推定できると考えた．平均悪手とレート値の関係が一次関数になると仮定し，探索深さ 11 の Bonanza での平均悪手の結果から最小二乗法を用いて以下のように求めた．

$$\text{レート値} = -3148 \times \text{平均悪手} + 4620 \tag{1}$$

このときのレート値とは，インターネット将棋対局サイトの「将棋倶楽部 24」でのレート値である．また，AI プレイヤの探索深さを変えることで AI プレイヤの強さを変化させた場合でも，精度に誤差はあるものの AI プレイヤ自身よりも強いプレイヤも実力を判断できる可能性が高いことが示された．

本研究との類似点は，AI プレイヤを用いて実力を推定している点とゲームの行動記録から実力を推定している点である．異なる点としては，NN やエラーレートを用いていない点，2 人確定完全情報ゲームを題材としている点である．

3　基礎知識

3 章では，麻雀，行動記録を集めたインターネット麻雀サイト，AI プレイヤ，エラーレート，実験で用いた手法について述べる．

3.1　麻雀

3.1.1　ルール概要

麻雀は牌を用いて遊ぶ 4 人不確定不完全情報ゲームである．牌はマンズ，ピンズ，ソーズ，字牌の 4 つの牌種に分けられる．マンズ，ピンズ，ソーズはそれぞれ 1 から 9 までの 9 種である．字牌は東，南，西，北，白，發，中の 7 種である．全部で牌は 34 種からなる．牌は各種 4 枚ずつあり，合わせて 136 枚である．マンズ，ピンズ，ソーズのことをまとめて数牌と呼ぶ．本研究では，数牌それぞれの 5 の牌 1 枚ずつを赤牌と呼ばれるボーナス牌に変更したルールを扱う．

各プレイヤは牌 13 枚 (手牌) を持ち，順番に牌を 1 枚引いて不要な牌を 1 枚捨てること (打牌) を繰り返す．その手牌を所持するプレイヤにしか見えない情報である．各プレイヤは牌山と呼ばれる場所から順番に牌を引いてくる．牌山の牌は各プレイヤが引くまでどのプレイヤにもそれが何の牌かはわからない．あるプレイヤが手牌を特定の形にそろえ，役が 1 つ以上あるとき，そのプレイヤは得点する．これを和了と呼ぶ．役とは手牌のそろえ方や状況によって成立するものである．牌山からすべての牌を引いてもだれも和了しない場合を流局と呼ぶ．誰かが和了または流局するまでを局と呼ぶ．麻雀では複数回の局を行い，最終的な持ち点からプレイヤの順位が決まる．本研究では，基本的な局数が 8 局である東南戦を扱う．

基本的な手牌のそろえ方は 4 面子 1 雀頭を作ることである．面子とは同じ牌種の牌 3 枚で構成される．面子には順子と刻子の 2 種類がある．順子とはマンズの 123 やピンズの 678 のように数牌が 3 つ順に並んだものである．刻子とはピンズの 222 やソーズの 555 といった同じ牌を 3 つそろえたものである．雀頭とは同じ牌を 2 つそろえたものである．

麻雀の行動選択肢には打牌以外に副露と呼ばれるものがある．副露にはポン，チー，アンカン，ダイミンカン，カカンの 5 種類がある．副露とは手牌の中の面子を確定させて，全プレイヤに公開する行為である．ポンとは，自分が 2 枚以上持つ牌を誰かが捨てたときに行え，捨てられたその牌をもらうことでその牌 3 枚を刻子の形で確定し，他の全プレイヤに公開する．チーとは，左隣のプレイヤから捨てられた牌をもらうことで順子を確定し，その順子を他の全プレイヤに公開する．ポンまたはチーをした場合，牌をもらった後に手牌から 1 枚牌を捨てる．アンカンとは，手牌で同じ牌を 4 枚持っているときに行うことができ，その 4 枚を刻子として確定し，他の全プレイヤに公開する．ダイミンカンとは，同じ牌を 3 枚持っていて，その牌を誰かが捨てたときに行うことができる．その牌 4 枚を刻子として確定し，他の全プレイヤに公開する．カカンとは，自分がポンしている牌をさらに持っているときに行うことができ，ポンしている牌にその牌を加えた 4 枚で刻子を確定する．アンカン，ダイミンカン，カカンを行った後には嶺上と呼ばれる牌山とは異なる場所から牌を 1 枚引いて，その後 1 枚捨てる．また，アンカン，ダイミンカン，カカンを行ったあとにはドラと呼ばれるボーナス牌が増える．

麻雀の特殊な行動選択肢として，リーチというものがある．リーチとは，1,000 点を払い，手牌が後 1 枚でそろうことと手牌をこれ以上入れ替えないことを他プレイヤに宣言する行動である．リーチと同時に打牌を行う．これによって役が 1 つ成立する．払われた 1,000 点は次に和了したプレイヤが獲得する．また，アンカンを除く副露を行ったプレイヤは，そのプレイヤはリーチすることができない．

表 3.1.1 は，麻雀のゲーム進行においてプレイヤの可能な行動を表にしたものである．プレイヤの可能な行動は打牌，副露，リーチ，和了のいずれかである．表 2 はプレイヤが行動可能なタイミングとその行動を表にしたものである．副露やリーチは行動可能なタイミングかつ条件がそろったときにしか行えない．表 3 は，麻雀のゲーム状況を表すために必要な要素の一覧である．

表 1: プレイヤの可能な行動選択肢

打牌	手牌から牌を 1 枚捨てる
ポン	他プレイヤの捨てた牌をもらい，刻子を確定
チー	他プレイヤの捨てた牌をもらい，順子を確定
アンカン	手牌の同じ牌 4 枚を刻子として確定
ダイミンカン	他プレイヤの捨てた牌をもらい，同じ牌 4 枚を刻子として確定
カカン	ポンして確定した刻子にもう 1 枚同じ牌を加えて刻子として確定
リーチ	あと 1 枚で手牌 14 枚がそろい，手牌を入れ替えないと他プレイヤに宣言
和了	手牌 14 枚がそろい得点

表 2: プレイヤの意思決定点

意思決定点	可能な行動
牌を引いたとき	打牌，アンカン，カカン，リーチ，和了
正面，右プレイヤが打牌したとき	ポン，ダイミンカン，和了
左プレイヤが打牌したとき	チー，ポン，ダイミンカン，和了

表 3: あるプレイヤ視点でのゲーム状況の構成要素

手牌	和了するためにそのプレイヤがそろえる牌
捨て牌	プレイヤ 4 人が打牌した牌
副露牌	プレイヤ 4 人が副露した確定面子
ドラ表示牌	ボーナス牌であるドラを決める牌
親番	局の最初に牌を引くプレイヤ
持ち点	プレイヤそれぞれの点数
リーチ状況	プレイヤそれぞれのリーチ有無
局数	現在のゲーム進行度
本場	親番が何局連続しているか
供託	まだどのプレイヤも獲得していない，リーチによって払われた点数

3.1.2 天鳳

天鳳とはインターネット麻雀サイトの 1 つである [7]．累計登録ユーザー数は 480 万人以上，180 日以内にプレイしているアクティブユーザー数は 34 万人以上であり，最もよく遊ばれているインターネット麻雀サイトの 1 つである．天鳳にはプレイヤの実力指標として，段位とレート値がある．段位は対戦結果に応じて上下し，対戦で得られるポイントが一定以上貯まると昇段，失うと降段する．段位は新人からスタートし，9 級，8 級，…，初級，初段，二段，…，十

段，天鳳位と上昇していく．レート値は対戦結果と相手プレイヤのレート値に応じて上下する．レート値の初期値は 1,500 である．レートの変動値の計算式は以下の通りである．

$$(\text{レートの変動値}) = (\text{対戦数補正}) \times (\text{対戦結果} + \text{補正値}) \tag{2}$$

対戦数補正は 400 戦以上では 0.2 で固定値である．400 戦未満の場合では以下の式で計算する．

$$(\text{対戦数補正}) = 1 - \text{対戦数} \times 0.002 \tag{3}$$

対戦結果は 1 位 + 30, 2 位 + 10, 3 位 – 10, 4 位 – 30 となる．補正値は次のように計算する．

$$(\text{補正値}) = (\text{卓の平均レート値} - \text{自分のレート値})/40 \tag{4}$$

天鳳では，プレイヤの段位とレート値によって対戦できる場所が限定されている．段位 7 段以上かつレート値 2,000 以上のプレイヤは鳳凰卓，4 段かつレート値 1,800 以上のプレイヤは特上卓，1 級以上のプレイヤは上級卓，それ以外は一般卓で打つことができる．

天鳳では，麻雀の行動記録である牌譜を生成している．プレイヤは対戦を行うと，自分の対戦の牌譜を得られる．自分の対戦以外の牌譜は鳳凰卓の牌譜のみ公開されており，容易に手に入れることができる．しかし，鳳凰卓以外の牌譜は一般に公開されていない．有志のプレイヤが牌譜を提供するような場もないため，初級者中級者の牌譜を収集することは困難である．

3.1.3 AI プレイヤ

本研究では「Ako_Atarashi」という AI プレイヤを用いる．Ako_Atarashi は栗田萌氏が作成した AI プレイヤである [8][9][10]．ゲーム状況を与えると，可能な行動の価値を推定し，推定した価値が最も大きな行動を行う AI プレイヤである．推定する価値は，1 位 +90，2 位 +30，3 位 –30，4 位 –90 と順位点が与えられるとしたときの順位点の期待値である．天鳳では，プレイヤは通常，段位におけるポイント期待値を最大化するようにプレイするため，Ako_Atarashi と天鳳プレイヤの最大化したい価値は異なる．

Ako_Atarashi は，AI プレイヤの「爆打」と同等の実力を持つ AI プレイヤである．爆打は天鳳において，一般的に上級者の指標の 1 つとされる 7 段レート 2,000 を達成した AI プレイヤであり，人間の上級者と同等以上の実力を持つ [11][12]．つまり Ako_Atarashi も人間の上級者と同等以上の実力を持つ十分に強い AI プレイヤである．

3.2 エラーレート

エラーレートとは，バックギャモンプレイヤの実力指標の 1 つである．1 行動あたりの推定価値の減少平均値を表しており，値が 0 に近いほどそのプレイヤは優秀であるといえる．エラーレートは行動記録と十分な実力をもつ AI プレイヤから計算されるものである．AI プレイヤは，得点期待値 (価値) を推定するために用いられる．

バックギャモンの商用プログラムの 1 つである eXtreme Gammon が行うエラーレートの計算方法を以下に示す [13]．バックギャモンプレイヤの意思決定過程をマルコフ決定過程として考える [14]．非終端状態の集合を S，状態 $s \in S$ における行動集合を $A(s)$ とする．複数の行動記録から状態 s と行動 a の対 (s, a) がいくつも採取でき，対の数が N 個であったとする．n 番目の状態行動対は (s_n, a_n) と書く．n は $1 \leq n \leq N$ である．状態と行動の対の集合 D は以下のように表す．D には複数の行動記録から採取した対が属している．

$$D = \{(s_1, a_1), \ldots, (s_N, a_N)\} \tag{5}$$

状態 s において行動 a を選択したときの eXtreme Gammon が推定する価値を $Q(s,a)$ とする．エラーレート $e(D)$ の計算式は以下のように表す．

$$e(D) = \frac{500}{|D|} \sum_{(s,a)\in D} \max_{a'\in A(s)} Q(s,a') - Q(s,a) \tag{6}$$

ただし 1 ポイントマッチの場合のエラーレートはさらに 1.5 倍する．

eXtreme Gammon の推定する価値は，ゲームの設定によって異なる．アンリミットの場合では得点期待値，ポイントマッチの場合ではゲーム終了時の勝率である．D に含まれる状態行動対の採取は駒の移動，ダブルの提案，ダブルのテイク選択のときに行う．状態行動対を採取する条件は，$|A(s)| \geq 2$ かつ $\max_{a\in A(s)} Q(s,a) - \min_{a\in A(s)} Q(s,a) \geq 0.001$ である．また，ダブルの提案では，提案する前の推定価値の方が相手がテイクした後より 0.2 以上高い場合，ダブルを提案する前の推定価値の方が相手がパスしたときより 0.2 以上高い場合，ダブルを提案する前も相手がテイクした後も推定価値がとても低い場合においては状態行動対を採取しない．

3.3 ニューラルネットワーク (NN)

NN とは生物の神経回路の構造を模倣して作られたネットワークで，多くのデータから特徴を推定する [15]．ネットワークの学習では，損失関数というものを用いて重みの調整を行い，目的の出力を得る．NN は多くのデータから特徴を推定することができ，特に人間が表現することが難しい特徴について推定することができる．NN をいくつも積み重ね，多層化することでより複雑な情報処理をさせる．NN には層というものがある．層は図 1 における縦にユニット (図 1 における丸) が並んでいる部分のことである．層には入力層，中間層，出力層と 3 つの種類がある．図 1 の場合，$l = 1$ の層が入力層，$l = 2$ の部分が中間層，$l = 3$ の部分が出力層である．

NN の基本的なものとして，順伝播型ネットワークがある．ネットワークは図 1 のような構造である．層状に並べたユニットが隣り合う層の間でのみ枝でつながっていて，情報が入力側から出力側にのみ伝播する NN である．枝の部分を重みと呼ぶ．順伝播型ネットワークでは，ユニット間の重みが一部なかったり重みの値を共有することがある．

次の層のユニットへの入力は前の層のユニット出力と重みによって計算される．例えば図 1 の $l = 2$ 層の一番下のユニットへの入力 $u_3^{(2)}$ は，$l = 1$ 層のユニットの出力を上から順に $z_1^{(1)}, \cdots, z_5^{(1)}$ とすると次のように計算される．

$$u_3^{(2)} = z_1^{(1)} w_{13}^{(2)} + z_2^{(1)} w_{23}^{(2)} + z_4^{(1)} w_{23}^{(2)} \tag{7}$$

ユニットへの入力に対して活性化関数と呼ばれる関数を作用させ，この層のユニットの出力として次の層に値を渡す．活性化関数にはシグモイド関数や ReLU 関数などが用いられる．例えば図 1 の $l = 2$ 層の一番下のユニットからの出力 $z_3^{(2)}$ は，活性化関数 f を用いて，次のように計算される．

$$z_3^{(2)} = f(u_3^{(2)}) \tag{8}$$

シグモイド関数 $f(x)$ は次のようなものである．

$$f(x) = \frac{1}{1 + \mathrm{e}^{-x}} \tag{9}$$

シグモイド関数は x の値が小さいほど 0 に近い値をとり，x の値が大きいほど 1 に近い値をとる．

ReLU 関数 $g(x)$ は次の式で表される．

$$g(x) = \max(x, 0) \tag{10}$$

ReLU 関数を用いると，計算量が小さく，学習がより早く進むという利点がある．

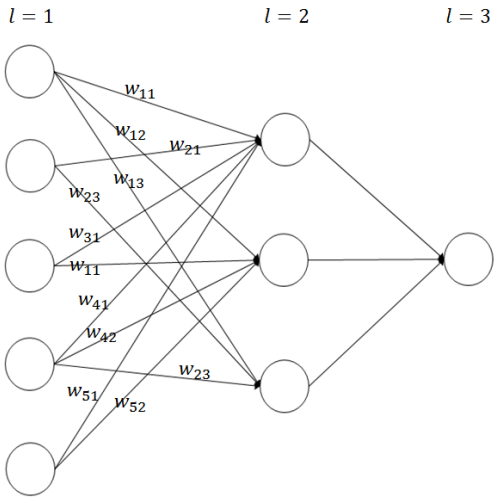

図 1: 順伝播型ネットワーク

前の層のユニットすべてとつながれているユニットからなる層を全結合層と呼ぶ．第 l 層のユニットの出力を $z_1^{(l)} \cdots z_n^{(l)}$，$l+1$ 層のユニットへの入力を $u_1^{(l+1)} \cdots u_m^{(l+1)}$，ユニット $x_i^{(l)}$ と $u_j^{(l+1)}$ を結ぶ重みを $w_{ij}^{(l+1)}$ とすると，バイアスという一つの値 $b_j^{(l+1)}$ を用いて $u_j^{(l+1)}$ は

$$u_j^{(l+1)} = z_1^{(l)} w_{1j}^{(l+1)} + z_2^{(l)} w_{2j}^{(l+1)} + \cdots + z_n^{(l)} w_{nj}^{(l+1)} + b_j^{(l+1)} \tag{11}$$

と表される．全結合層のネットワーク例としては図 2 のようなネットワークである．

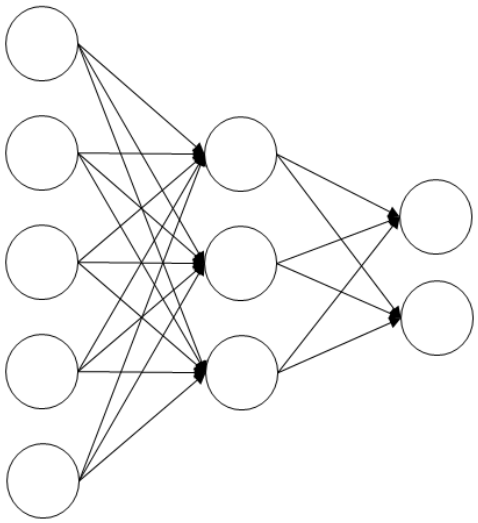

図 2: 全結合層

3.4 重みの学習

ネットワークの学習は大きく分けると，教師あり学習と教師なし学習の 2 種類がある．教師あり学習では，入力と共に出力すべき値である目標出力というものも与えて出力と目標出力の

誤差が小さくなるように学習させる．教師なし学習では，入力のみを与えて学習させ機械が自ら分類などを行う．基本的な NN では，図 1 のように多くの枝が伸びていて学習していく過程でどの枝が重要でどの枝が重要でないか，つまり重みを学習していく．重みを調整することで目的の出力を得る．重みを調整するにあたって，損失関数 (目的関数) というものを用いる．損失関数はどれくらい学習がうまくいっているかを表していて，0 に近いほど学習がうまくいっているといえる．損失関数にはソフトマックス関数や交差エントロピー，二乗誤差などが用いられる．本研究では，二乗誤差を 2 で割ったものを損失関数に用いた．二乗誤差 E を表す式は以下の通りである．

$$E = \sum_{i=1}^{N} \|d_i - g_i(x)\|^2 \tag{12}$$

N は出力のユニット数，d_i は i 番目の目標値，$g(x_i)$ は入力 x に対するネットワークの i 番目の出力を表す．

順伝播型ネットワークの教師あり学習において学習を行う場合，出力と目標出力の誤差が小さくなるように重みとバイアスを更新する必要がある．そのために勾配降下法がよく用いられる．この方法では，誤差が小さくなる方向に向かって少し重みを調整し，これを何度も繰り返す．更新式としてはつぎのように表す．

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \gamma \nabla E(\boldsymbol{w}^{(t)}) \tag{13}$$

$\boldsymbol{w}^{(t+1)}$ は調整した後の重み，$\boldsymbol{w}^{(t)}$ は調整する前の重み，γ は学習係数で 0 より大きい十分小さな値，$\nabla E(\boldsymbol{w}^{(t)})$ は勾配であり $\nabla E(\boldsymbol{w}^{(t)}) = \partial E(\boldsymbol{w}^{(t)})/\partial \boldsymbol{w}^{(t)}$ である．上の式より，$\boldsymbol{w}^{(t+1)}$ の勾配 $E(\boldsymbol{w}^{(t+1)})$ はテーラー展開を用いて次のように計算できる．

$$\begin{aligned} E(\boldsymbol{w}^{(t+1)}) &= E(\boldsymbol{w}^{(t)} - \gamma \nabla E(\boldsymbol{w}^{(t)})) \\ &\simeq E(\boldsymbol{w}^{(t)}) + \frac{\partial E(\boldsymbol{w}^{(t)})}{\partial \boldsymbol{w}^{(t)}}(-\gamma \nabla E(\boldsymbol{w}^{(t)})) + R \\ &= E(\boldsymbol{w}^{(t)}) - \gamma \nabla E(\boldsymbol{w}^{(t)})^2 + R \end{aligned} \tag{14}$$

R はテーラー展開における第 3 項目以降の項の和である．γ が十分小さいとき，R は無視できる．すると，γ は正の値であることから次の関係式が導ける．

$$E(\boldsymbol{w}^{(t+1)}) \leq E(\boldsymbol{w}^{(t)}) \tag{15}$$

このことから，上記の更新式で重みを調整することで誤差関数を小さくすることができる．

勾配降下法のうち，よく使われる手法が確率的勾配降下法 (stochastic gradient descent) であり頭文字をとって SGD と呼ばれる．SGD では，全学習データのうち一部だけを用いて計算を行い重みとバイアスを更新する．次の更新を行うときには異なる学習データを用いて計算を行う．勾配降下法の更新式ではすべての学習データに対しての誤差関数 $E(\boldsymbol{w})$ の勾配 E を用いていたが，SGD の更新式ではある学習データ n について計算される誤差関数 $E_n(\boldsymbol{w})$ の勾配 E_n を用いて次のように重みの更新を行う．

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \gamma \nabla E_n \tag{16}$$

$E(\boldsymbol{w})$ と $E_n(\boldsymbol{w})$ の関係は，学習データの数を N とすると次のようである．

$$E(\boldsymbol{w}) = \sum_{n=1}^{N} E_n(\boldsymbol{w}) \tag{17}$$

SGD は，全ての訓練データを用いて重みとバイアスの更新を行う場合と比べていくつか利点がある．例えば，訓練データが冗長性を持つ場合，計算効率が上昇する．また訓練データすべてを

使う場合，局所解につかまってしまうと抜け出せないという欠点があるが，SGD では毎回異なる訓練データで更新を行うため，局所解から抜け出せる可能性がある．

重みやバイアスの更新の際，前回のステップの更新量を用いることで収束を早め，振動を抑える手法がある．前回のステップの更新量にあたる項を慣性項という．慣性項の係数 μ は 1.0 以下の正の実数である．慣性項を用いた重みの更新式は以下のようになる．

$$\boldsymbol{w}^{(t+1)} = \boldsymbol{w}^{(t)} - \epsilon \nabla E(\boldsymbol{w}^{(t)}) - \mu \nabla E(\boldsymbol{w}^{(t-1)}) \tag{18}$$

SGD の他にも重みとバイアスを調整する手法の一つとして，Adaptive moment estimation (Adam) がある [16]．Adam による重みの更新を例として以下に示す．以下に示す Adam の重み更新手順は元論文 [16] の内容を一部要約したものである．Adam は学習係数の他に 3 つのパラメータ β_1，β_2，ϵ を用いる．1 次モーメントベクトル $m_0 = 0$，2 次モーメントベクトル $v_0 = 0$，タイムステップ $t = 0$ とする．

1. タイムステップ t の数値を $t + 1$ にする．
2. タイムステップ t における重みの勾配 $\nabla E(\boldsymbol{w}^{(t)})$ をもとめる．
3. 1 次モーメントの概算値を次の式で更新する．

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) \nabla E(\boldsymbol{w}^{(t)}) \tag{19}$$

4. 2 次モーメントの概算値を次の式で更新する．$\nabla E(\boldsymbol{w}^{(t)})^2$ は $\nabla E(\boldsymbol{w}^{(t)})$ の各成分を二乗した数値を持つベクトルである．

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) \nabla E(\boldsymbol{w}^{(t)})^2 \tag{20}$$

5. 1 次モーメントのバイアス補正を行った推定値を次の式でもとめる．

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{21}$$

6. 2 次モーメントのバイアス補正を行った推定値を次の式でもとめる．

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{22}$$

7. タイムステップ t における重みを次式で更新する．

$$\boldsymbol{w_t} = \boldsymbol{w_{t-1}} - \gamma \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} \tag{23}$$

8. 1 に戻る．

このような手順で各パラメータの更新を行う．

勾配降下法や Adam を行うには，誤差関数 $E(\boldsymbol{w})$ の勾配 $\nabla E = \partial E(\boldsymbol{w}) / \partial \boldsymbol{w}$ を計算する．ベクトルの各成分は各層での重みの誤差関数の微分 $\partial E / \partial w_{ij}$ とバイアスの誤差関数の微分 $\partial E / \partial b_j$ である．これらの計算は入力層に近いほど後ろの層の重みやバイアスも微分する必要があり，計算量が大きくなってしまう．誤差逆伝播法では，この微分を効率よく計算することができる．

$l-1$ 番目の層のユニット i と l 番目の層のユニット j をつなぐ重み $w_{ij}^{(l)}$ による誤差関数の微分は次のように表せる．

$$\frac{\partial E}{\partial w_{ij}^{(l)}} = \frac{\partial E}{\partial u_i^{(l)}} \frac{\partial u_i^{(l)}}{\partial w_{ij}^{(l)}} \tag{24}$$

この式の右辺の $\frac{\partial u_i^{(l)}}{\partial w_{ij}^{(l)}}$ はユニット i からの出力 $z_i^{(l-1)}$ に変形できる．さらに右辺の $\frac{\partial E}{\partial u_i^{(l)}}$ を $\delta_j^{(l)}$ とすると次のようになる．

$$\frac{\partial E}{\partial w_{ij}^{(l)}} = \delta_j^{(l)} z_i^{(l-1)} \tag{25}$$

ここで用いる $\delta_j^{(l)}$ は次のように求められる．

$$\delta_j^{(l)} = \sum_k \delta_k^{(l+1)} (w_{kj}^{(l+1)} f'(u_j^{(l)})) \tag{26}$$

ここでの $f'(u_j^{(l)})$ は活性化関数 f を $u_j^{(l)}$ で微分したものである．このため，$\delta^{(l)}$ は $l+1$ 層のユニットの $\delta^{(l+1)}$ によって計算できる．出力層を L 番目の層とすると，出力層での $\delta^{(L)}$ は次のように計算できる．

$$\delta_j^{(L)} = \frac{\partial E}{\partial u_j^{(L)}} \tag{27}$$

これは選んだ誤差関数によって計算方法は異なるが，ネットワークの出力と目標出力の値からもとめることができる．これらから，出力層側から入力層側に伝播することで目的の重みやバイアスに対して微分を行うことができ，勾配降下法を行うことができる．全結合層における誤差逆伝播法の計算式は，先ほどの式と同様で

$$\frac{\partial E}{\partial w_{ij}^{(l)}} = \delta_j^{(l)} z_i^{(l-1)} \tag{28}$$

となる．

3.5 自己符号化器

自己符号化器 (autoencoder) とは，教師なし学習によってデータをよく表す特徴を得るために用いる NN である．自己符号化器によって入力に対する特徴を自動抽出することができる．また NN における重みの初期値をよりよいものに設定するためにも用いられる．自己符号化器の学習では，図 3 のように入力層と出力層のユニット数を同じにする．中間層は 1 層のみで，基本的には入力層のユニット数よりも中間層のユニット数を少なくする．このようなネットワークを調整して，入力と出力が近くなるようにネットワークの重みとバイアスの学習を行う．自動抽出された入力に対する特徴は，中間層の出力値である．

自己符号化器は中間層が一つだけではなく複数の中間層を持つ場合もあり，これを多層自己符号化器という．その場合，多層の NN を出力層で折り返すことで作成する．図 4 は多層自己符号化器の例である．用いる多層の NN は，通常出力層に向かって層のユニット数が少なくなっている．中間層が 1 層の自己符号化器よりもより複雑な表現ができるが，その分学習が困難である．

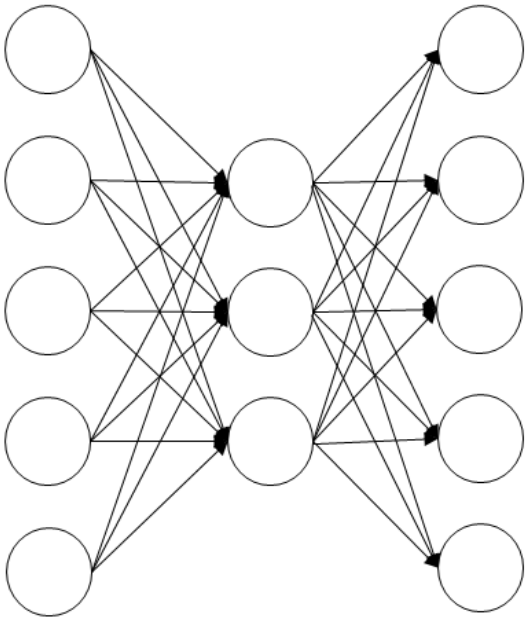

図 3: 自己符号化器のネットワーク

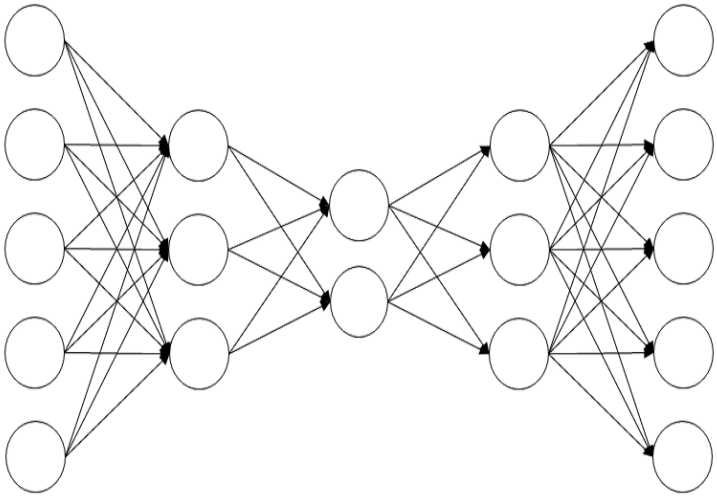

図 4: 多層自己符号化器のネットワーク例

4 目的

本研究の目的は，麻雀プレイヤの実力を少ない行動記録から推定することである．現状，麻雀の実力指標としては平均順位やレート値が用いられるが，おおまかにプレイヤの実力を推定しようとすると 500 程度の対戦が必要である．例えば麻雀の対戦を 500 戦行うとすると，通常 250 時間以上かかってしまう．そこで，少ない行動記録から実力を推定しようと考えた．本研究では，2 つのアプローチから麻雀プレイヤの実力の推定を行い，その性能を評価する．

1 つ目はエラーレートから実力を推定する方法である．AI プレイヤを用いてエラーレートを計算し，実力推定を行う．また，その性能を平均順位から実力を推定する場合と比較する．

2 つ目は NN を用いて実力を推定する方法である．NN を用いて行動記録から様々な特徴を獲得し，より少ない行動記録から実力の推定ができないか調査する．特に，1 戦の行動記録から実力を推定することを目指した．

推定性能の評価には相関係数と決定係数を用いた．相関係数や決定係数を計算するには実力を表す指標が必要である．しかし，麻雀プレイヤの実力を表す指標を得ることは困難である．本研究で行動記録を収集した天鳳では，3.1.2 節で述べた通り，プレイヤのレベルごとに対戦できる場所が分かれているため，平均順位はプレイヤの真の実力の指標として適切ではない可能性がある．一方，天鳳でのレート値は，他プレイヤのレベルによってレート変動値が変化するため，プレイする場所が異なっていてもある程度実力を表していると考えられる．そのため本研究では，500 戦以上にわたって平均化されたレート値 (平均レート) をそのプレイヤの最も信頼できる実力の指標として扱った．

5　エラーレートを用いた実力の推定

この章では，あるプレイヤの行動記録と AI プレイヤを用いて，そのプレイヤのエラーレートを計算し，プレイヤの実力を推定する実験について述べる．本実験でのデータ 1 つは，n 戦分のあるプレイヤ p の行動記録から得られたエラーレート，その n 戦分の p の平均順位，その n 戦を含む 500 戦以上から得られた p の平均レート R_p の組である．収集した行動記録は，天鳳での東南戦喰いタンあり赤ありのルールの行動記録に対応する．

本実験では，エラーレートと平均順位のプレイヤの実力の推定性能を比較した．性能の評価は，天鳳の平均レートとの相関係数，決定係数により行った．

5.1　実験方法

この節では，エラーレート，相関係数，決定係数の計算方法について説明する．

本実験では，Ako_Atarashi の推定する価値を用いてエラーレート $e(D)$ を計算する．計算方法は 3.2 節で紹介した eXtreme Gammon での計算方法とほぼ同じである．異なる点は，推定価値 $Q(s,a)$ が Ako_Atarashi が推定する価値である点，状態行動対の採取の仕方が Ako_Atarashi が推定価値を出力する場合には $|A(s)| \geq 2$ であるときに状態行動対を採取し，推定価値を出力しない場合では採取を行わない点，AI プレイヤの推定価値に関わらず採取を行う点，計算式が以下の式となる点である．

$$e(D) = \frac{1}{|D|} \sum_{(s,a)\in D} \max_{a'\in A(s)} Q(s,a') - Q(s,a) \tag{29}$$

エラーレートと平均レートの相関係数と決定係数の計算について述べる．採取した状態行動対の全体集合を $D_A = \{D_1, \ldots, D_P\}$ と書く．ただし $D_p = \{D_{p1}, \ldots, D_{pM_p}\}$，$D_{pi}$ はプレイヤ p の行動記録から得られた i 番目の状態行動対の集合で，n 戦分の対からなる．さらに $M = \sum_{p=1}^{P} M_p$，M_p はプレイヤ p のデータ数とする．

$e(D_{pi})$ のデータ全体にわたって計算された平均値を e_{ave} とする．式は以下の通りである．

$$e_{\text{ave}} = \frac{1}{M} \sum_{p=1}^{P} \sum_{i=1}^{M_p} e(D_{pi}) \tag{30}$$

M_p を考慮した R_p の重み付き平均値 R_{ave} は以下のように計算される．

$$R_{\text{ave}} = \frac{1}{M} \sum_{p=1}^{P} M_p R_p \tag{31}$$

エラーレート $e(D_{pi})$ と平均レート R_p の相関係数は以下の式で計算する．

$$\text{相関係数} = \frac{\frac{1}{M}\sum_{p=1}^{P}\sum_{i=1}^{M_p}(e(D_{pi}) - e_{\text{ave}})(R_p - R_{\text{ave}})}{\sqrt{\frac{1}{M}\sum_{p=1}^{P}\sum_{i=1}^{M_p}(e(D_{pi}) - e_{\text{ave}})^2}\sqrt{\frac{1}{M}\sum_{p=1}^{P}\sum_{i=1}^{M_p}(R_p - R_{\text{ave}})^2}} \tag{32}$$

相関係数の値は，推定値と目標値の正の相関が大きいほど 1 に近づき，負の相関が大きいほど −1 に近くなる．相関係数の値が 0 に近いほど，目標値にかかわらず無作為に推定していることを表す．

決定係数の計算式は以下の式で計算する [17]．ただし関数 f はエラーレートから平均レートを推定する関数である．

$$\text{決定係数} = 1 - \frac{\sum_{p=1}^{P}\sum_{i=1}^{M_p}(R_p - f(e(D_{pi})))^2}{\sum_{p=1}^{P}\sum^{M_p}(R_p - R_{\text{ave}})^2} \tag{33}$$

決定係数の値は，推定が理想的な場合 1 となる．値が 0 の場合では目標値の平均値を出力する推定と同程度であることを表し，値が負の場合では目標値の平均値を推定する場合よりも悪い推定性能であることを表す．

平均順位と平均レートの相関係数，決定係数を計算する場合には，式 (32)， (33) のエラーレートを平均順位に，エラーレートの平均値 e_{ave} を平均順位の平均値に，関数 f を平均順位から平均レートを推定する関数に置き換えて相関係数，決定係数の計算を行う．

5.2 実験結果

この節では，エラーレートと平均順位の性能評価の実験結果を示す．本節の表に示す不確かさは標準誤差から見積もられた 95%信頼区間である．

天鳳の一般卓から鳳凰卓まですべてのレベルでの牌譜を用いてエラーレートを計算した．その中で，500 戦以上対戦しているプレイヤは 13 人であった．13 人の試合数は合計で 7,934 であった．本実験のデータ全体は，この 13 人の誰かが対戦した牌譜から構成した．表 4 はデータ全体の統計情報である．Ako_Atarashi のレート値は 2000 以上であることが予想されるので，Ako_Atarashi は表 4 の 13 人の誰よりも実力が高いと考えられる．

表 4: データ全体の統計情報

p	対戦数	平均値を推定した標本の大きさ	平均レート	平均順位	エラーレート
1	1177	533	1969.9	2.49	0.247
2	2802	580	1936.2	2.49	0.277
3	680	674	1896.7	2.39	0.231
4	529	529	1863.1	2.45	0.282
5	715	536	1833.5	2.40	0.275
6	1065	736	1732.9	2.47	0.292
7	3128	549	1726.3	2.45	0.315
8	1820	673	1710.1	2.46	0.314
9	6024	803	1700.3	2.52	0.336
10	1033	505	1695.3	2.48	0.377
11	1114	562	1571.6	2.53	0.466
12	12085	658	1553.3	2.50	0.475
13	626	596	1453.6	2.67	0.486

エラーレートと平均レートの関係が一次関数になると仮定し，500 戦以上の行動記録から得られたエラーレートと平均レートから最小二乗法を用いて関係式 34 を求めた．

$$\text{レート値} = -1648.2 \times \text{エラーレート} + 2296.0 \tag{34}$$

同様に，平均順位と平均レートの関係が一次関数になると仮定し，500 戦以上の平均順位と平均レートから最小二乗法を用いて関係式 (35) を求めた．

$$\text{レート値} = -1580.8 \times \text{平均順位} + 5669.7 \tag{35}$$

500 戦以上のエラーレートと平均レートの散布図，500 戦以上の平均順位と平均レートの散布図を図 5 に示す．横軸が平均レート，左縦軸が 500 戦以上のエラーレート，右縦軸が 500 戦以上の平均順位を表す．赤の点がエラーレートと平均レートの関係をプロットした点であり，緑の点が平均順位と平均レートの関係をプロットした点である．また，赤の直線はエラーレートと平均レートの関係式 (34) であり，緑の直線は平均順位と平均レートの関係式 (35) である．エラーレート，平均順位ともに平均レートが高くなると小さくなる傾向が確認できた．

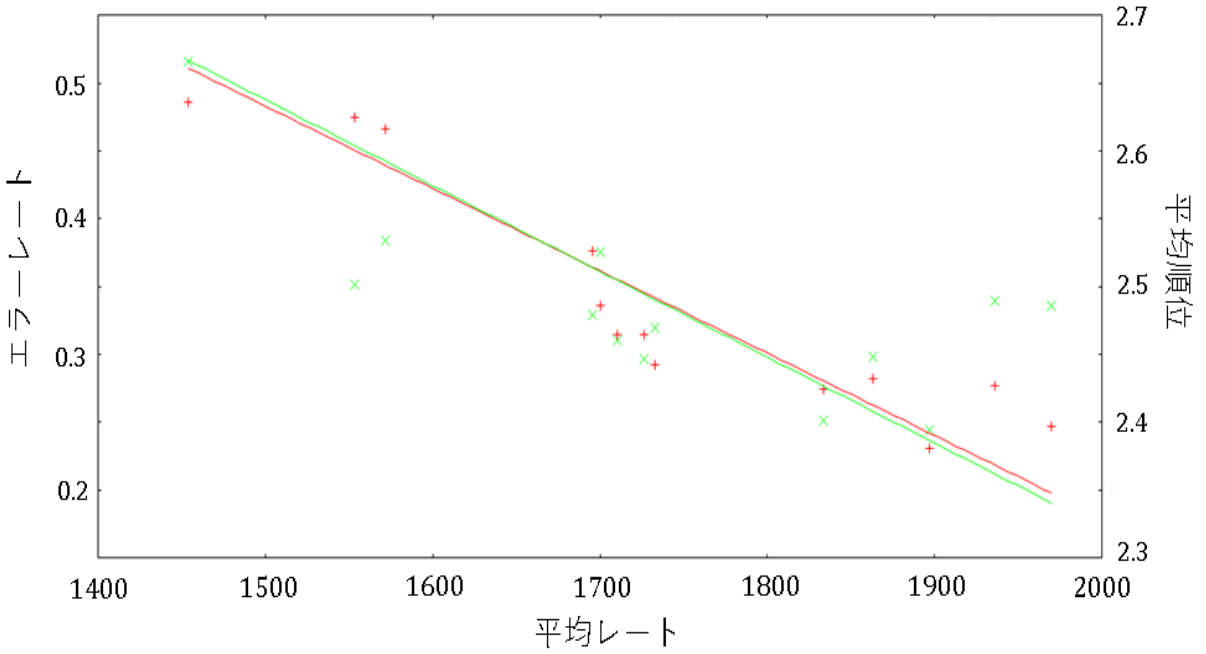

図 5: エラーレート，平均順位と平均レートの散布図

表 5 は，n 戦分の行動記録から得られたエラーレートと平均レートから相関係数と決定係数を求めた結果である．決定係数は関係式 (34) を用いて計算を行った．対戦数 n が大きくなるに従って相関が大きくなることが確認できた．対戦数 1 でもエラーレートと平均レートの間に弱い相関があることが確認でき，対戦数 16 では強い相関があると言える結果となった．対戦数が 4 以下の場合では，決定係数が負の値になっていることから式 (34) による推定はあまりうまくいっていない．対戦数 n が大きくなるに従い，決定係数が大きくなっていることから式 (34) による推定がある程度うまくいっていると考えられる．

平均順位と平均レートの関係は表 6 の通りである．決定係数は関係式 (35) を用いて計算を行った．表 6 から，対戦数 32 の平均順位をとっても平均レートとの相関はほとんど確認できなかった．そのため，行動記録が少ない場合では，平均順位は実力を表していないと考えられる．対戦数 500 の場合では，誤差範囲が大きいものの平均順位はある程度実力を推定できていると考えられる．決定係数は，対戦数が 500 以外では負の値となっていることからうまく推定できていない．

また，表 5 の 16 戦での相関係数と表 6 の 500 戦での相関係数を比較すると，誤差の範囲となってはいるものの 16 戦で計算したエラーレートの方が高い相関係数となっている．また，決

表 5: エラーレートと平均レートの相関

対戦数 n	相関係数	決定係数
1	-0.374 ± 0.023	-11.44
2	-0.49 ± 0.03	-1.42
4	-0.59 ± 0.03	-0.40
8	-0.69 ± 0.04	0.15
16	-0.76 ± 0.05	0.41
32	-0.78 ± 0.06	0.51
64	-0.82 ± 0.07	0.62
128	-0.84 ± 0.09	0.66
500	-0.93 ± 0.07	0..87

表 6: 平均順位と平均レートの相関

対戦数 n	相関係数	決定係数
1	-0.041 ± 0.022	-142.89
2	-0.06 ± 0.03	-71.77
4	-0.08 ± 0.04	-35.77
8	-0.12 ± 0.06	-16.71
16	-0.17 ± 0.09	-8.08
32	-0.23 ± 0.12	-3.85
64	-0.30 ± 0.16	-1.67
128	-0.41 ± 0.21	-0.47
500	-0.7 ± 0.3	0.48

定係数で比較した場合でも同等程度の結果となった．これらの結果から，16 戦のエラーレートからの推定精度は，500 戦の平均順位からの推定精度と同程度であると考えられる．

6　NNを用いた実力の推定

この章では，NNを用いて，5章の実験よりもさらに少ない対戦数からプレイヤの実力推定を試みる実験について述べる．この章でのデータ1個とは，1戦分のあるプレイヤの行動記録から得られたエラーレート，その1戦分の状態行動対，そのプレイヤのレート値の組である．収集した行動記録は，天鳳での東南戦喰いタンあり赤ありのルールの行動記録である．また，本研究では，構築したNNの層間は基本的に全結合されている．

6.1　エラーレートとNNを用いた実験

この節では，1戦分の行動記録から計算したエラーレート1つを入力とするNNを用いてプレイヤの実力を推定を試みた実験について述べる．

6.1.1　実験方法

平均レートは収集できたプレイヤ数が少ないため持ちず，レート値を目標値としてNNの学習を行った．NNの学習に用いた訓練データは16,656個，テストデータは10,310個用意した．あるプレイヤの行動記録から採取したデータは，訓練データまたはテストデータのどちらかに一方にのみ含まれるようにした．

本実験で構築したNNをNNeと呼ぶこととする．NNeへの入力は1戦の行動記録から計算したエラーレートの値であり，出力はレート推定値である．NNeの構成は，入力層のユニット数が1個，中間層のユニット数が16個，出力層のユニット数が1個の3層NNである．調整する重みの数はバイアスの数も含めて49個である．活性化関数にはReLU関数を用いた．最適化手法にはSGDを用いた．学習係数γは10^{-7}である．慣性項を用い，その係数μを0.9とした．NNeの推定するレート値と採取したデータのレート値の誤差を計算する損失関数には，平均二乗誤差を2で割ったものを用いた．

6.1.2　実験結果

図6はNNeの学習に用いたデータすべてでのエラーレートとレート値の関係を散布図にしたものである．横軸がレート値，縦軸がエラーレートを表している．レート値が1500である部分をみると，他のレート値と比べて，点が集中しているように見える．これは天鳳のレート値の初期値が1500であることが影響していると考えられる．図6から，エラーレートとレート値に弱い相関があるように見える．エラーレートとレート値の相関はNNeの結果と共に表7に後述する．

図7にNNeの学習結果を示す．学習回数と損失関数値の関係を表している．横軸が学習回数，縦軸は損失関数の値である．緑のグラフが訓練データでの損失関数の値，紫のグラフがテストデータでの損失関数の値を表している．学習が進むに従い，損失関数の値が減少していくことが確認できた．また，訓練データでの損失関数の値は18,000前後に，テストデータでの損失関数の値は20,000前後の値となった．訓練データのレート値の平均値を出力した場合，テストデータでの損失関数の値は20,200程度であるため，少なくとも平均値を出力するのと同程度の性能になっていると考えられる．

表7はテストデータでのレート値とNNeの推定値との関係とNNeの学習に用いたデータすべてでのエラーレートとレート値の関係をまとめたものである．エラーレートとレート値の決

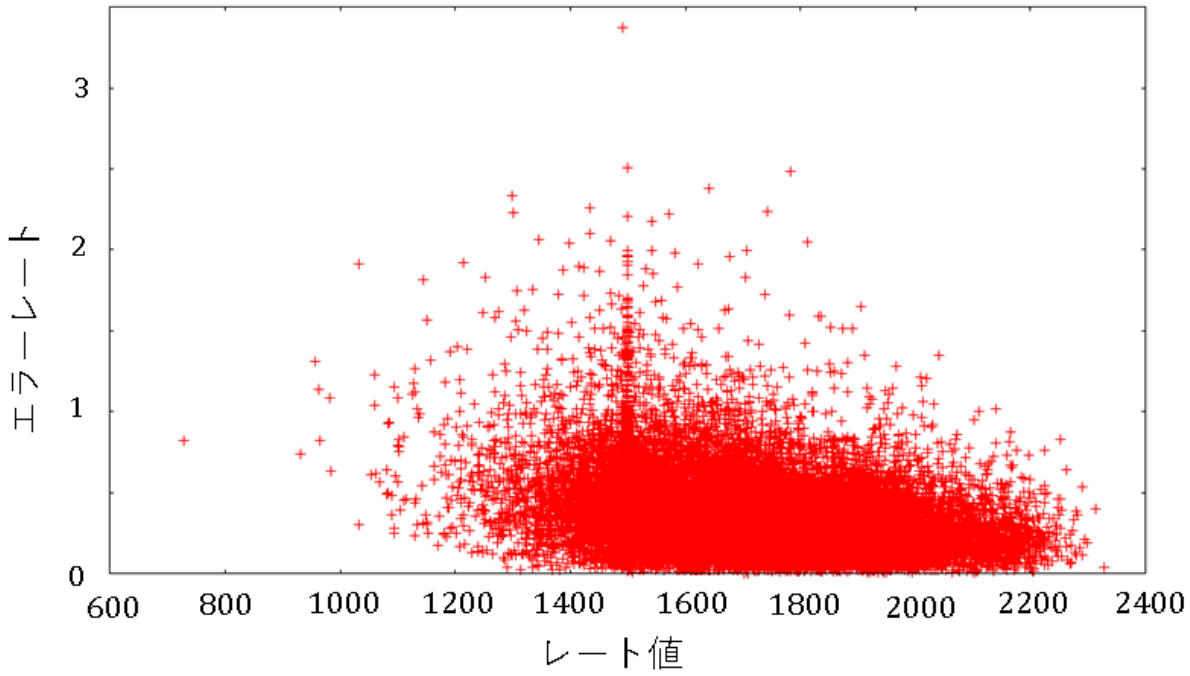

図 6: エラーレートとレート値の散布図

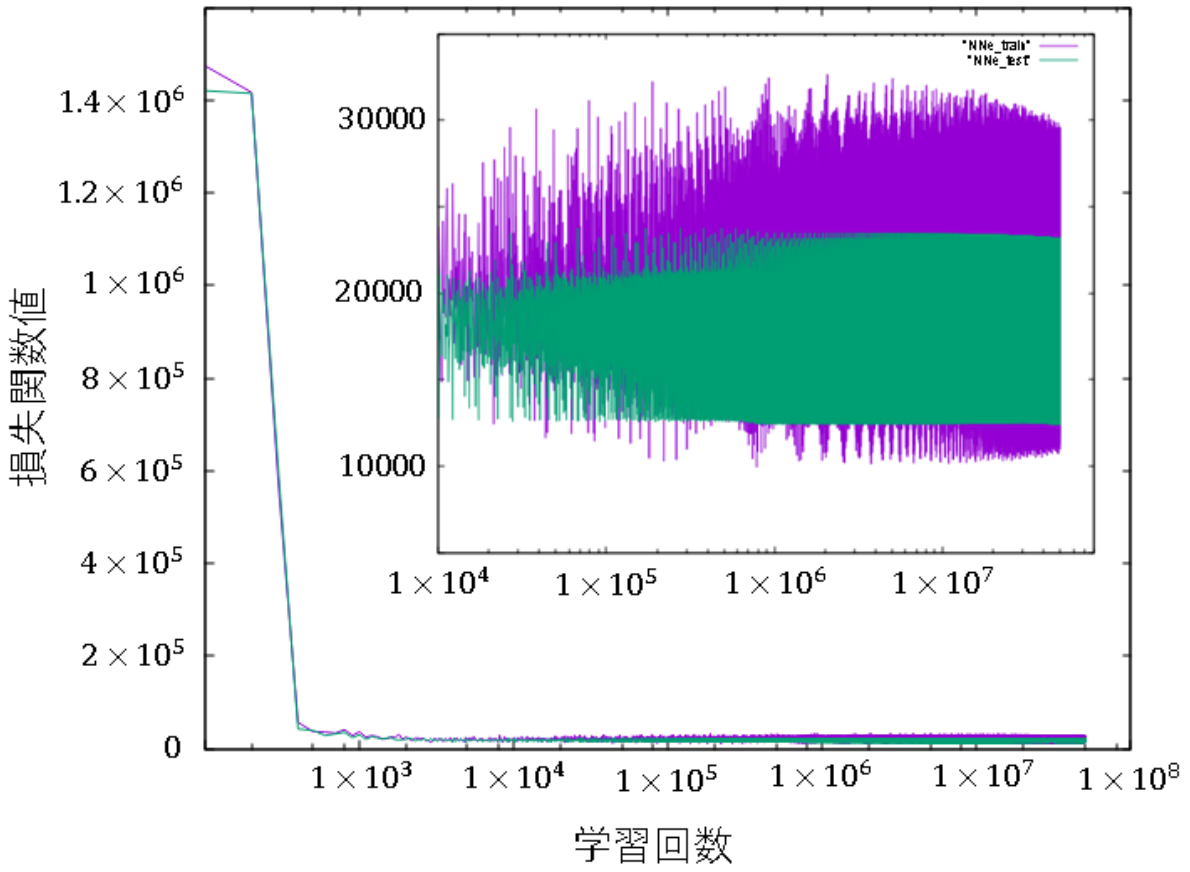

図 7: 損失関数値 (NNe)

定係数は式 (34) を用いて計算したものである．相関係数はエラーレートから推定した場合と，NNe の推定値には差が見られなかった．NNe の推定値は決定係数が 0 程度であり，ランダムに推定しているのと変わらない結果となった．NNe の出力値を確認したところ，出力のほとんどが 1,600 から 1,750 程度の値を出力しており，実力がかなり低いであろうプレイヤや，実力がかなり高いであろうプレイヤを判断することはできていなかった．訓練データの平均値が約 1,700 であることから，平均値付近を多く出力していると考えられる．

表 7: NNe の推定性能

	相関係数	決定係数
NNe の推定値	0.315 ± 0.017	0.03
エラーレート	-0.314 ± 0.011	-0.84

6.2 自己符号化器を用いた特徴の抽出

この節では，自己符号化器を用いて麻雀の状態行動対から特徴の抽出を試みる実験について述べる．1 行動分の行動記録から自己符号化器を用いて特徴を抽出できないか実験を行った．

6.2.1 実験方法

本実験では，データ 1 つを状態行動対 1 つとする．自己符号化器の学習に用いた訓練データは 14,556,362 個，テストデータは 2,000,000 個用意した．データは，上級者初級者にかかわらず行動記録から採取した状態行動対である．本実験では，行動記録から得られる状態行動対 1 つを自己符号化器への入力とする．復元した状態行動対 1 つが自己符号化器の出力である．1 つの状態行動対を表 8 のように数値化する．状態行動対 1 つは計 4913 個の数値で表現した．自分の手牌と各プレイヤの持ち点以外は 01 で表した．手牌は 0 から 4 の整数，持ち点は 0 から 100,000 の整数で表現した．ただし自己符号化器に入力する際には，手牌と持ち点の数値化部分は平均 0，分散 1 になるように標準化を行った．

表 8: 状態行動対の数値化

特徴の種類	特徴の概要	特徴数
行動選択肢	打牌，手出しか，副露した種類，リーチ，副露しない	11
打牌した牌	打牌した牌の種類	37
副露の詳細	副露した牌，副露されたプレイヤ	$37 + 4$
自分の手牌	手牌の構成	37
捨て牌	捨てたプレイヤ，捨てた牌	$(4 + 37) \times 86$
副露状況	副露したプレイヤ，副露されたプレイヤ， 副露の種類，副露した牌	$(4 + 4 + 7 + 37) \times 20$
リーチ状況	全員のリーチ状況	4
その他の状況	現在の局，本場，供託，ドラ表示牌，持ち点	$12 + 8 + 8 + 37 \times 5 + 4$

本実験では，2 種類のネットワーク構成で実験を行う．1 つ目は 3 層 NN の自己符号化器である．2 つ目は多層化した自己符号化器であり，5 層 NN で自己符号化器を設計した．3 層の自己符号化器を AE，5 層の自己符号化器を MAE と呼ぶこととする．AE のネットワークの構成は，入力層，中間層，出力層の 3 層である．AE の中間層のユニット数を l とする．MAE のネットワークの構成は，入力層，中間層 1，中間層 2，中間層 3，出力層の 5 層である．中間層 1 と 3 のユニット数を 256，中間層 2 のユニット数を l' とする．l および l' を $1, 2, 4, \ldots$ と変化させて特徴抽出の性能を比較する．調整する重みの数はバイアスの数も含めて，AE は $9,827l + 4,913$ 個，MAE は $513l' + 2,520,881$ 個である．

活性化関数には ReLU 関数を用いた．最適化手法には Adam を用いた．学習係数 γ は 2×10^{-6} である．Adam のパラメータをそれぞれ，$\beta_1 = 0.9$，$\beta_2 = 0.99$，$\epsilon = 10^{-8}$ とした．数値化した状態行動対と AE および MAE で復元した状態行動対の誤差を計算する損失関数には平均二乗誤差を 2 で割ったものを用いた．

6.2.2 実験結果

AE と MAE を学習させた結果，損失関数の値は図 8 から図 15 のようになった．図はすべて，横軸は学習回数，縦軸は損失関数の値を表している．図 8 はテストデータでの，AE の l が $1, 4, 16, 64, 256, 1024$ の場合の損失関数値のグラフである．上のグラフから順に l が $1, 4, 16, 64, 256, 1024$ のグラフに対応する．図 9 はそれを拡大したものである．図 10 はテストデータでの，AE の l が $2, 8, 32, 128, 512$ の場合の損失関数値のグラフである．上のグラフから順に l が $2, 8, 32, 128, 512$ のグラフに対応する．図 11 はそれを拡大したものである．

l にかかわらず，学習が進むに従い損失関数値が減少していく様子が確認できた．テストデータの平均値を出力した場合，損失関数値は 50 程度であることから $l = 1$ または 2 の場合では，平均値を出力した場合と同程度の性能であると考えられる．また，l が増えるに従い，損失関数値が小さくなっていることが確認できた．l が 4,8,16,32 の場合では，まだ損失関数値の減少が続いているようであるが，$l = 64$ のグラフと同程度かそれ未満の性能であると予想される．

図 12 はテストデータでの，MAE の l' が $1, 4, 16, 64$ の場合の損失関数値のグラフである．上のグラフから順に l' が $1, 4, 16, 64$ のグラフに対応する．図 13 はそれを拡大したものである．図 14 はテストデータでの，MAE の l' が $2, 8, 32, 128$ の場合の損失関数値のグラフである．上のグラフから順 l' が $2, 8, 32, 128$ のグラフに対応する．図 11 はそれを拡大したものである．

l' に関わらず，学習が進むに従い損失関数値が減少していく様子が確認できた．$l' = 1$ の場合では損失関数値が 50 程度であることから，平均値を出力した場合と同程度の性能であると考えられる．また，l' が増えるに従い，損失関数値が小さくなっていくことが確認できた．l' が 1 以外の場合ではどれも収束しきっていないようであるが，時間の都合上ここまでしか結果を確認できなかった．

表 9 は，現時点で確認できた AE，MAE の損失関数値を表にまとめたものである．現時点では，MAE の l' が $2, 4, 8, 16$ のとき，$l = l'$ である AE よりも性能がよいという結果となった．しかし，l' が 32 よりも大きい場合では，$l = l'$ である AE のほうが MAE よりも性能がよいという結果となっている．これはネットワークの学習が収束しきっていないため，このような結果となったと考えられる．MAE の学習が十分であれば，$l = l'$ である AE と少なくとも同等以上の性能となると予想する．

AE，NNAE ともに，損失関数値がほとんど 0 になり，特徴抽出がほぼ完全にできたという結果は得られなかったが，行動記録をある程度表す特徴を獲得できたと考えられる．

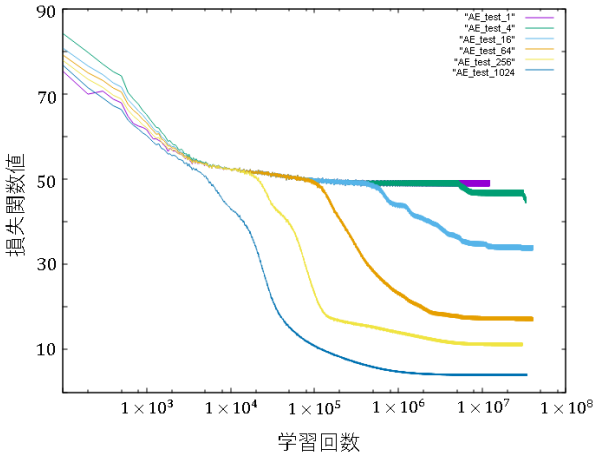

図 8: 損失関数値 1 (AE)

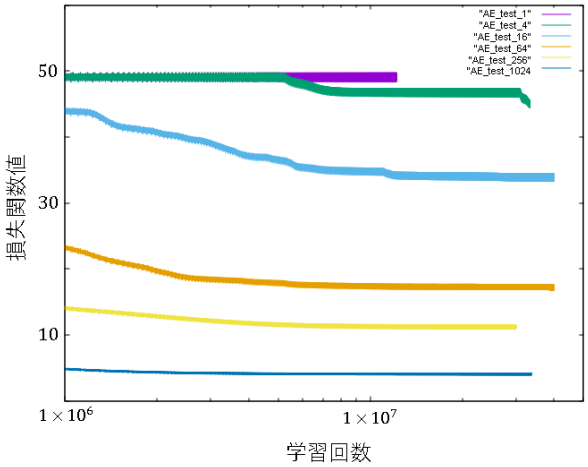

図 9: 損失関数値 1 (AE 拡大)

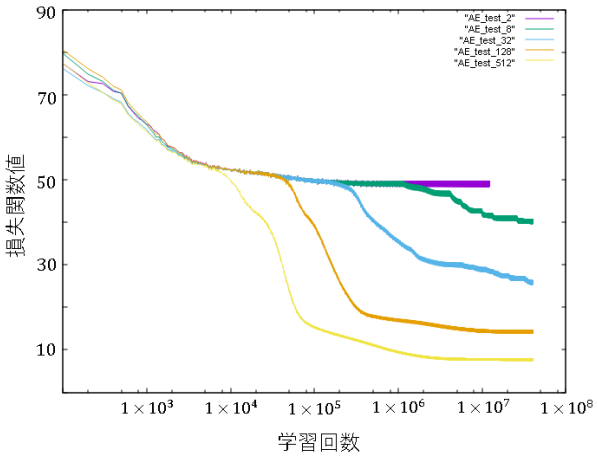

図 10: 損失関数値 2 (AE)

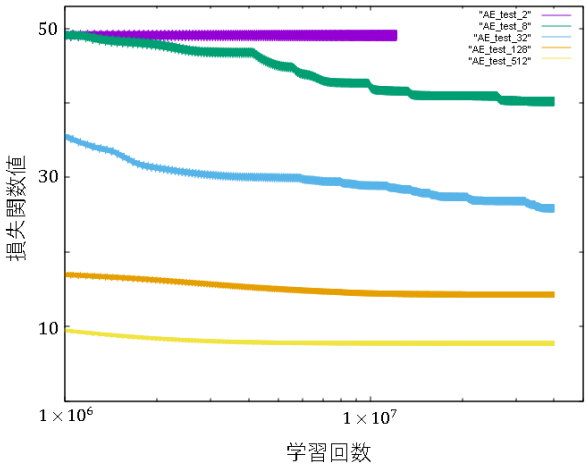

図 11: 損失関数値 2 (AE 拡大)

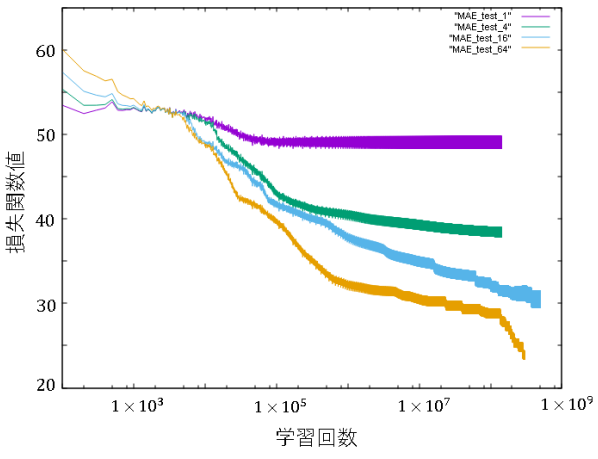

図 12: 損失関数値 1 (MAE)

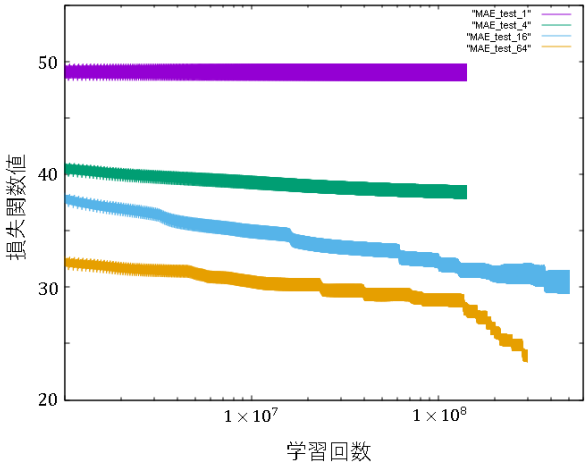

図 13: 損失関数値 1 (MAE 拡大)

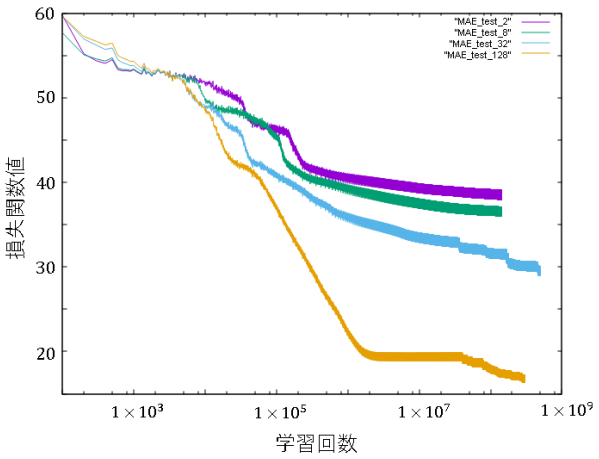

図 14: 損失関数値 2 (MAE)

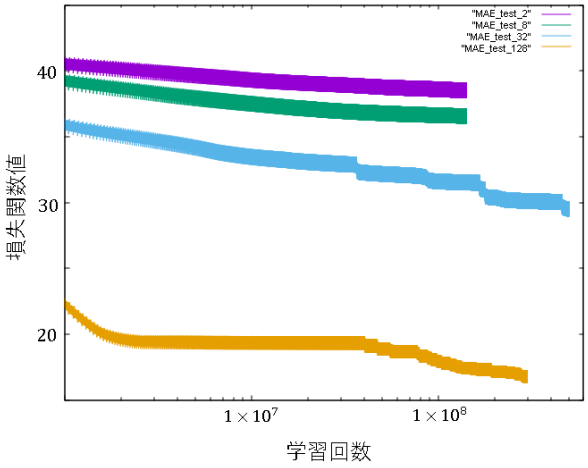

図 15: 損失関数値 2 (MAE 拡大)

表 9: AE と MAE の損失関数値

p と p'	AE の損失関数値	MAE の損失関数値
1	49	49
2	49	39
4	45	38
8	40	37
16	34	30
32	26	29
64	17	24
128	14	17
256	11	
512	8	
1024	4	

6.3 抽出した特徴を用いた実力の推定

この節では，節 6.2 で作成した AE を用いてプレイヤの実力を推定する実験について述べる．

6.3.1 実験方法

NN の学習に用いた訓練データは 15,114 個，テストデータは 11,867 個である．あるプレイヤの行動記録から採取したデータは，訓練データまたはテストデータのどちらか一方にのみ含まれるようにした．

本実験で構築した NN を NNAE と呼ぶこととする．NNAE への入力は 1 戦分の行動記録の状態行動対である．出力はレート推定値である．1 局での最大行動選択回数を 36，1 戦の最大局数を 22 とした．1 戦の符号化は，行動記録から状態行動対が採取できた場合では表 8 に従い数値化し，状態行動対が採取できない場合では 4,913 個をすべて 0 とした．NNAE への入力は $36 \times 22 \times 4,913$ 個の特徴である．NNAE の構成は図 16 のようになっている．層 A は入力 4,913 個，出力 256 個，層 B は入力 256 個，出力 4 個，層 C は入力 4 個，出力 1 個，層 D は入力 792

個，出力 1 個である．同じ名前の層は重みを共有している．層 A の重みの初期値を 6.2 節で学習した AE (中間層のユニット数 256) の重みに設定する．層 A の重みは更新しない．また，層 D の入力では，4,913 個の特徴をすべて 0 とした入力から計算された値が D への入力となる場合には，その入力を 0 となるようにする．調整する重みの数はバイアスの数を含めて 1,826 個である．

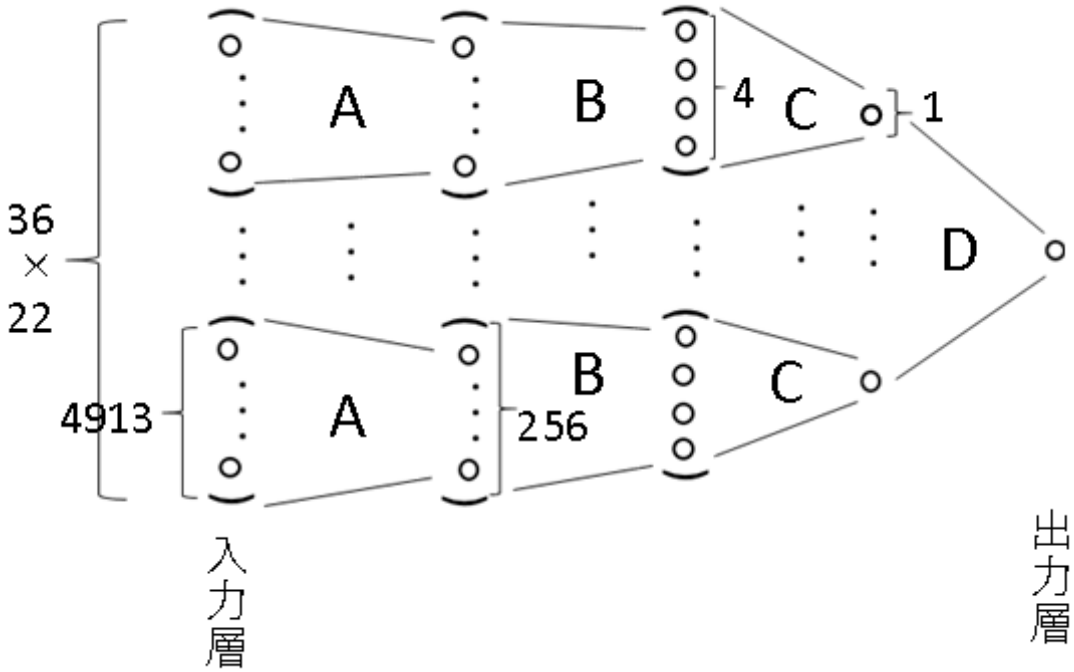

図 16: NNAE の構成

活性化関数には ReLU 関数を用いる．最適化手法には Adam を用いる．学習係数 γ を 10^{-3}，Adam のパラメータをそれぞれ，$\beta_1 = 0.9$，$\beta_2 = 0.99$，$\epsilon = 10^{-8}$ とした．NNAE の推定するレート値と採取したデータのレート値の誤差を計算する損失関数には，平均二乗誤差を 2 で割ったものを用いた．

6.3.2 実験結果

図 17 に NNAE の学習結果を示す．学習回数と損失関数値の関係を表している．横軸が学習回数，縦軸は損失関数の値である．緑のグラフが訓練データでの損失関数の値，紫のグラフがテストデータでの損失関数の値を表している．右上には学習回数 1×10^4 以上の結果を示す．学習回数が増えるに従い，損失関数の値が減少していくことが確認できた．訓練データでの損失関数の値は 19,000 程度，テストデータでの損失関数の値は 22,000 程度まで減少している．訓練データのレート値の平均値を出力した場合，損失関数の値は 22,000 程度であるので，平均値を出力する程度の性能だと考えられる．テストデータでのレート値と NNAE 推定値の相関係数と決定係数を計算したところ，表 10 のようになった．相関係数はほとんど 0 となっており，推定がうまくいっていない．また決定係数は負の値となり，平均値を出力する場合よりも悪い推定性能であるという結果となった．NNAE の推定値を確認したところ，ほとんど一定値に近い値を推定していた．

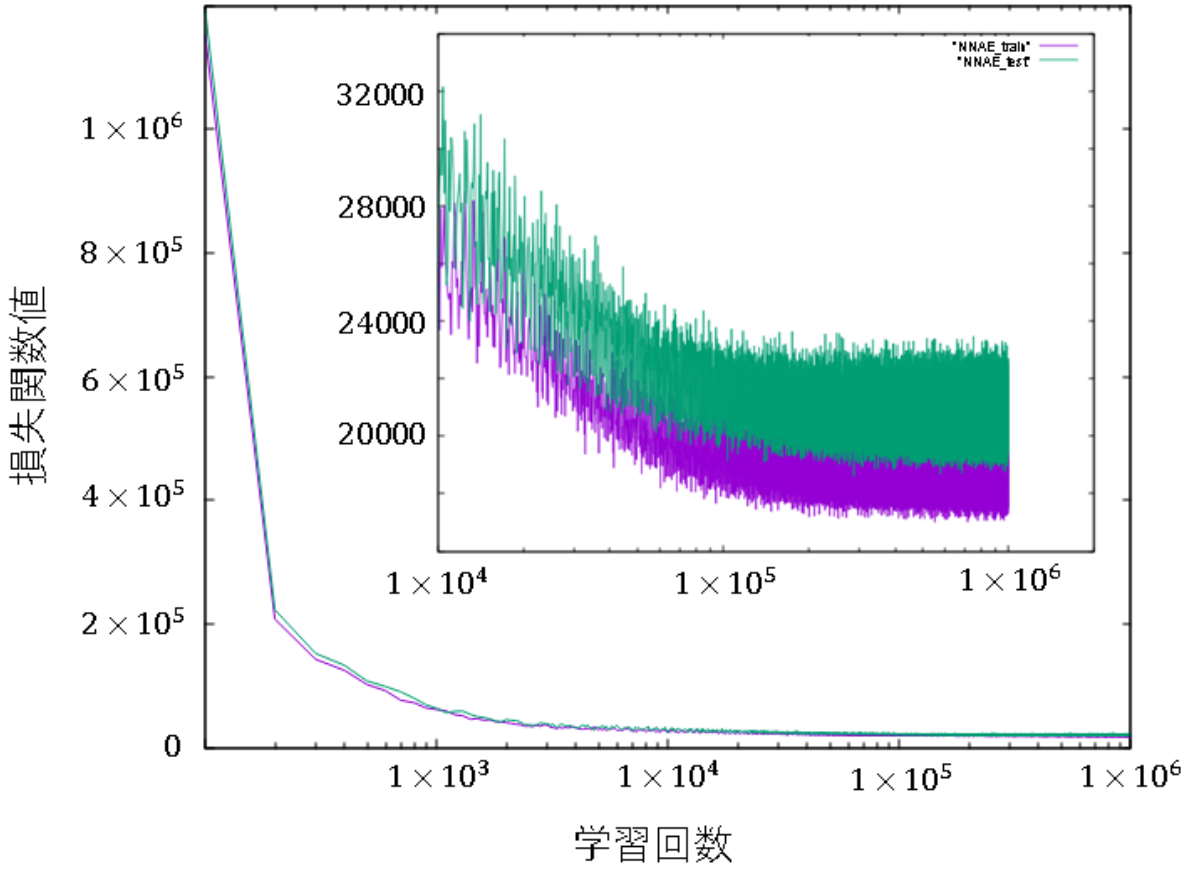

図 17: 損失関数値 (NNAE)

表 10: NNAE の推定性能

	相関係数	決定係数
NNe の推定値	0.004 ± 0.019	−0.11

6.4 抽出した特徴とエラーレートを用いた実力の推定

この節では，節 6.2 で抽出した特徴とエラーレートを組み合わせてプレイヤの実力を推定する実験について述べる．

6.4.1 実験方法

本実験で用いた訓練データは 16,523 個，テストデータは 10,457 個である．あるプレイヤの行動記録から採取したデータは，訓練データまたはテストデータのどちらか一方にのみ含まれるようにした．本実験で構築した NN を NNAEE と呼ぶこととする．NNAEE への入力は，行動記録 1 戦分の状態行動対とその行動のエラーレートである．出力はレート推定値である．NNAEE の構成は図 18 のようになっている．層 A は入力 4,913 個，出力 256 個，層 B は入力 258 個，出力 4 個，層 C は入力 4 個，出力 1 個，層 D は入力 792 個，出力 1 個である．同じ名前の層は重みを共有している．層 A の重みの初期値を 6.2 節で学習した AE (中間層のユニット数 256) の重みに設定する．層 A の重みは更新しない．層 B で追加される 2 個の入力は行動のエラーレート

とエラーレートの有無を表すフラグである．また，層 D の入力では，4,913 個の特徴をすべて 0 とした入力から計算された値が D への入力となる場合には，その入力を 0 となるようにする．調整する重みの数はバイアスの数を含めて 1,834 個である．

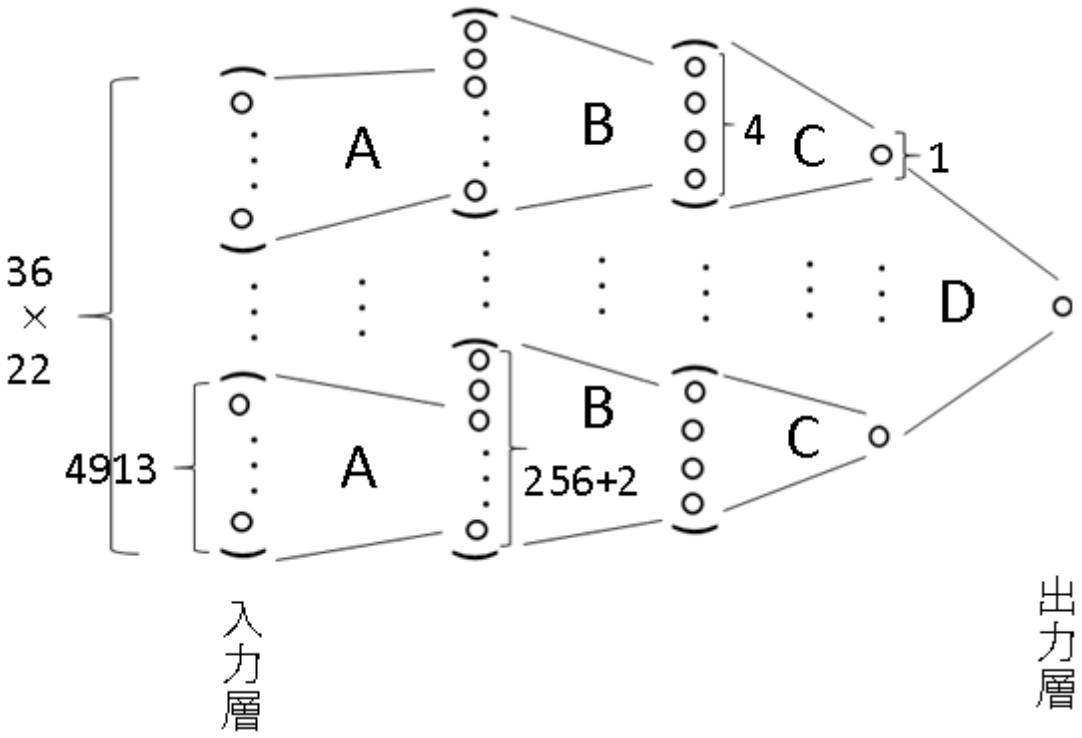

図 18: NNAEE の構成

活性化関数には ReLU 関数を用いる．最適化手法には Adam を用いる．学習係数 γ を 10^{-3}，Adam のパラメータをそれぞれ，$\beta_1 = 0.9$，$\beta_2 = 0.99$，$\epsilon = 10^{-8}$ とした．NNAEE の推定するレート値と採取したデータのレート値の誤差を計算する損失関数には，平均二乗誤差を 2 で割ったものを用いた．

6.4.2 実験結果

図 19 に NNAEE の学習結果を示す．学習回数と損失関数値の関係を表している．横軸が学習回数，縦軸は損失関数の値である．緑のグラフが訓練データでの損失関数の値，紫のグラフがテストデータでの損失関数の値を表している．右上には学習回数 1×10^4 以上の結果を示す．学習回数が増えるに従い，損失関数の値が減少していくことが確認できた．訓練データでの損失関数の値は 20,000 程度，テストデータでの損失関数は 22,000 程度まで減少している．訓練データのレート値の平均値を出力した場合，20,000 程度であるため平均値を出力するものと同程度か劣る性能であると考えられる．

テストデータでのレート値と NNAE 推定値の相関係数と決定係数を計算したところ，表 11 のようになった．相関係数はほとんど 0 となっており，推定がうまくいっていない．また決定係数は負の値となり，平均値を出力する場合よりも悪い推定性能であるという結果となった．NNAEE の推定値を確認したところ，ほとんど一定値に近い値を推定していた．平均値を出力した場合よりも劣る性能となっている原因として，学習回数が十分でなく収束しきっていないことが考えられる．

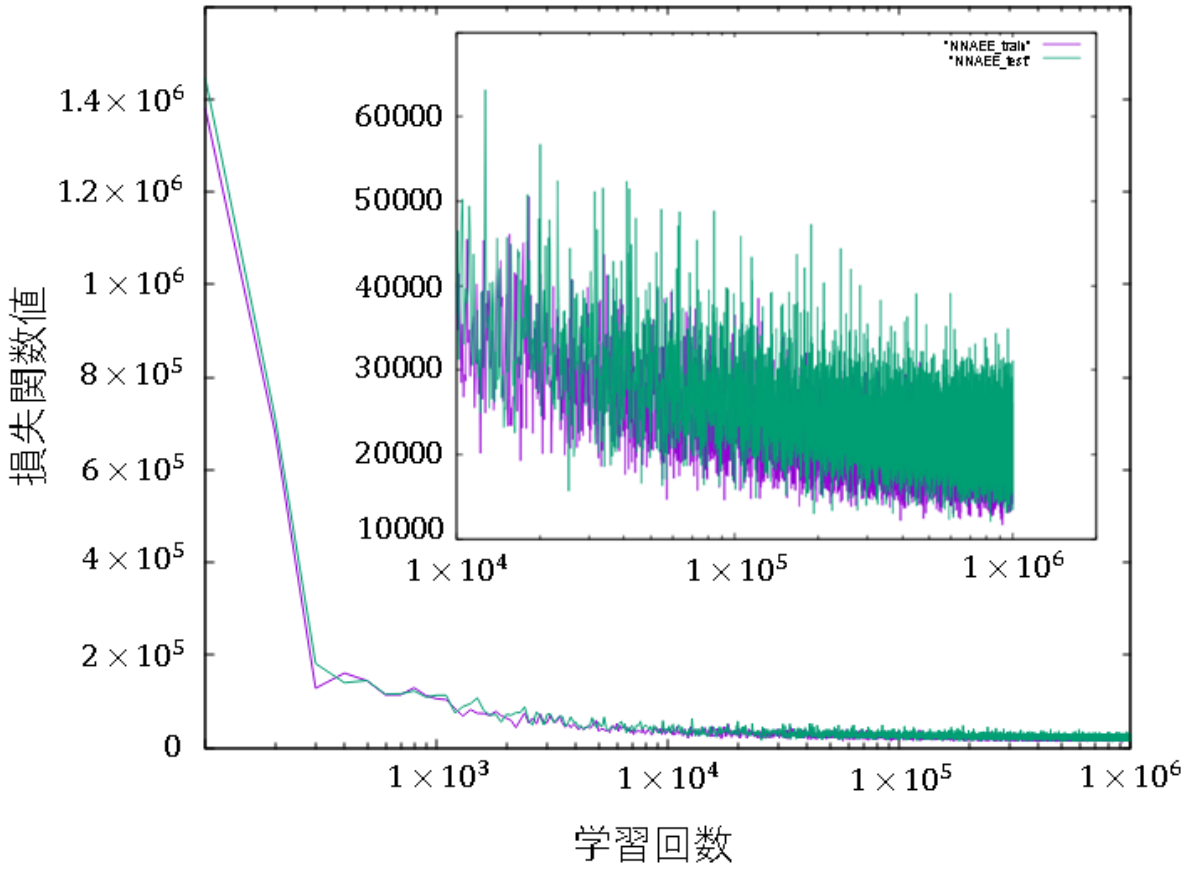

図 19: 損失関数値 (NNAEE)

表 11: NNAEE の推定性能

	相関係数	決定係数
NNe の推定値	−0.0018 ± 0.019	−0.17

7　おわりに

本研究では，少ない行動記録から麻雀プレイヤの実力を推定することを目指し，エラーレートを用いてプレイヤの実力を推定する実験と NN を用いてプレイヤの実力を推定する実験の 2 つを行った．

エラーレートを用いてプレイヤの実力を推定する実験では，対戦数 16 の行動記録から計算したエラーレートと平均レートに強い相関がみられた．また，16 戦の行動記録からもとめたエラーレートの推定性能と 500 戦の行動記録から求めた平均順位の推定性能が同程度という結果が得られ，現状の麻雀の実力指標よりも少ない行動記録から麻雀プレイヤの実力を推定できたと考えられる．

NN を用いてプレイヤの実力を推定する実験では，16 戦よりも少ない対戦数で実力を推定することを試みた．NN を用いて 1 戦から計算したエラーレート 1 つから実力を推定する実験では，NN を用いない場合と性能に差は見られなかった．自己符号化器を用いた行動記録から特徴を抽出する実験では，中間層のユニット数が増加するに従い，特徴抽出の精度はあがっていくことが確認できた．自己符号化器を多層化したところさらに特徴抽出がうまく行える可能性が示された．抽出した状態行動対の特徴から NN を用いてプレイヤの実力を推定することにも取り組んだが，平均値と同程度の性能しか得られなかった．

これらの結果を受けて，NN を用いてよりよいプレイヤの実力推定を行うためには，さらなる行動記録の収集や麻雀の行動記録の数値化の見直し，大きな規模の NN の構築などを行う必要がある．

謝辞

本研究を行うにあたり，指導教員である保木邦仁准教授には研究の方針から内容に至るまで多くのご指導をいただきました．心から感謝申し上げます．栗田萌氏には，AI プレイヤを貸与していただきました．株式会社 C-EGG からは牌譜を提供していただきました．心から感謝申し上げます．また，普段より助言を頂いた保木研究室，村松研究室，高橋研究室，西野研究室の皆様に感謝いたします．

参考文献

[1] とつげき東北．科学する麻雀．講談社，2014．

[2] 荒木伸夫，保木邦仁，村松正和．畳み込みニューラルネットワークを用いた囲碁における1局の棋譜からの推定．情報処理学会論文誌，Vol. 57，No. 11，pp. 2365-2373，2016．

[3] 荒木伸夫．囲碁に関する2つの情報工学的アプローチ．博士論文，電気通信大学，2017．

[4] Matej Guid，Ivan Bratko．Conputer Analysis of World Chess Champions．ICGA Journal，Vol. 2，No. 29，pp. 65-73，2006．

[5] Matej Guid，Ivan Bratko．Using Heuristic-Search Based Engins for Estimating Human Skill at Chess．ICGA Journal，Vol. 2，No. 34，pp. 71-81，2011．

[6] 山下宏．将棋名人のレーティングと棋譜分析．The 19th Game Programming Workshop 2014，pp. 9-16，2014．

[7] オンライン対戦麻雀 天鳳．url: http://tenhou.net．最終アクセス 2019．

[8] 栗田萌，保木邦仁．有効非巡回グラフで表現された1人麻雀の探索アルゴリズム．The 22nd Game Programming Workshop 2017，pp. 42-49，2017．

[9] 栗田萌，保木邦仁．麻雀1局の目的に応じた抽象化と価値推定からなるプレイヤの開発．The 22nd Game Programming Workshop 2017，pp. 72-79，2017．

[10] 栗田萌，保木邦仁．麻雀における他家の手牌と待ちの予測に基づく放銃率推定．情報処理学会研究報告，Vol. 2017-GI-38，No. 5，2017．

[11] 水上直樹，中張遼太郎，浦晃，三輪誠，鶴岡慶雅，近山隆．多人数性を分割した教師付き学習による4人麻雀プログラムの実現．情報処理学会論文誌，Vol. 55，No. 11，pp. 2410-2420，2014．

[12] 水上直樹，鶴岡慶雅．期待最終順位の推定に基づくコンピュータ麻雀プレイヤの構築．The 20nd Game Programming Workshop 2015，pp. 179-186，2015．

[13] eXtreme gammon．url: http://www.extremegammon.com/extremegammon2.pdf．最終アクセス 2019．

[14] Csaba Szepesvári，小山田創哲，前田新一，小山雅典．速習 強化学習 基礎理論とアルゴリズム．共立出版，2017．

[15] 岡谷貴之．深層学習．講談社，2015．

[16] Diederik Kingma，Jimmy Ba．Adam: A method for stochastic optimization．arXiv preprint arXiv:1412.6980，2014．

[17] Tarald O. Kvalseth．Cautionary Note about R^2．The American Statistician，Vol. 39，No. 4，pp. 279-285，1985．

[18] Caffe — Deep Learning Framework．url: http://caffe.berkeleyvision.org．最終アクセス 2019．