G-103

重み付け K 近傍法を用いた格闘ゲームにおける相手の行動予測と動的難易度調整

Online Adjustment of the AI's Strength in a Fighting Game Using the Weighted k-Nearest Neighbor Algorithm and a Game Simulator

中川 裕登†　　山本 界人†　　ターウォンマット ラック†
Yuto Nakagawa　Kaito Yamamoto　Ruck Thawonmas

1. はじめに

多くの対戦型ゲームにおいて、対戦相手の強さはゲームの楽しさや難易度に大きく影響している[1]. 対戦型格闘ゲームにおいて、個人のゲームに対する習熟性、行動決定などのユーザスキルに見合った相手と戦うことがゲームの楽しさの一つであるといえる. しかし、ユーザスキルによって求められる対戦相手の強さには違いがある.

多くの対戦型格闘ゲームでは、対人戦が主な遊戯目的だが AI とも対戦することができる. しかし、AI の強さはあらかじめ決められており、事前にルールベースで定められた行動をすることが主流である。AI の強さを動的に変更する手法としては、人間が操作するキャラクターの模倣と学習を通して強い AI を作る手法[2]や、模倣と学習を通してより賢く人間らしい AI を作る手法[3]などが存在する. しかし、これらの手法ではユーザスキルに見合った強さの AI を作ることはできない. AI との対戦をより楽しくするために、ユーザスキルに応じて動的に AI の強さを調整する必要がある.

本稿では、ユーザスキルに応じた AI の動的な強さの調整を目的とし、重み付け K 近傍法を取り入れた動的な難易度調整の手法を提案する. そして Fighting Game AI Competition のルールに基づいて、実験を行った結果から提案手法の評価を行う. 実験環境は Fighting Game AI Competition で用いるプラットフォームを用いる.

2. ゲームルール

本章では実験対象である Fighting Game AI Competition の大会ルールについて述べる. Fighting Game AI Competition が採用しているルールでは 1 ラウンド 60 秒とし、計 3 ラウンドを 1 試合とする格闘ゲームである. お互いが受けたダメージを用いてスコアを算出し、3 ラウンドの合計スコアが多い方が勝者となる. スコアは以下の式(1)によって求められる.

$$\text{SelfScore} = \frac{opp.roundDamage}{self.roundDamage+opp.roundDamage} * 1000 \quad (1)$$

ここで「roundDamage」は各キャラクターがそのラウンドで受けたダメージ量を表している。「roundDamage」の値は各ラウンドの開始時に 0 に初期化される。平均獲得スコアが 500 を上回っていれば、相手よりも勝っていると判断する。

3. 提案手法

重み付け K 近傍法を用いた AI を基本に、対戦相手のユーザスキルに応じて AI 自身の強さを抑えていくことで動的な難易度調整を行う。AI の強さを調整するために、累積スコア(以下、S_a)に応じて自分の行動を変化させる. S_aは 3 ラウンドを通して常に監視する. 強い AI を実現するために、提案手法では重み付け K 近傍法を用いて相手の行動を予測する. そして、AI 内部のシミュレータを用いて予測した相手の行動と自分の取れうる全ての行動の組み合わせでの 1 秒後までの行動結果をシミュレートする. なお、重み付け K 近傍法及び、シミュレータを用いた行動決定は次章で詳しく述べていく.

ゲーム中で自分が取ることができる行動をa_n(n=0, 1, 2, …n)とし、シミュレータ上でa_nを行った際の評価値を e[n]とする. n は行動の種類数を表している. e[n]はシミュレータ上で自分が与えたダメージと相手が与えたダメージとの差で求める. e[n]の値が負であればa_nは自分に不利な行動であり、正であれば有利な行動である.

自分の行動の決定方法は対戦状況の優劣に応じて行動決定を行うことで動的な難易度の調整を行う. そこで、優劣の判定をするための閾値（以下、τ)を設定する. τと現在のS_aを比較し、優劣の判定を行い、比較した結果に応じて行動決定の方法を変化させる. τよりS_aが低ければ、自分のスコアを上げるために e[n]の値が正で最大の行動を行う. 対して、τよりS_aが高ければ、自分のスコアを低く抑えるために、e[n]が負で絶対値が最小の行動を行う. 以上の提案手法をアルゴリズム 1 に記す.

Algorithm1 ActDecision(*self, opp, predictAct, game*)

```
e[i]⇦0 for all i = 1,2,…,n
S_a ⇐ 1000*(opp.gameDamage/
  (self. gameDamage + opp.gameDamage))
e ⇐ simulate(self, opp, predictAct, game)
if S_a ≧ τ then
   repeat
     d ⇐ argmin (|e|)
     if e[d] ≧ 0 then
        e[d] ⇐ ∞
     end if
   until e[d] < 0
else
    d ⇐ argmax (e)
end if
return a_d
```

self、opp は自身と相手のキャラクターを指し、キャラクターに関する情報を保持している. game はゲーム中に用いられるデータをまとめたものである. gameDamage は試合開始から自身が受けたダメージの総量を表している。そして simulate(self,opp,predictAct,game)を用いて各行動の e[n]を算出する. predictAct は重み付け K 近傍法で予測した相手の攻撃行動である。 dを用いて対戦状況に応じた最適な評価値を持つ行動a_dを決定する。

†立命館大学情報理工学部知能情報学科,
Information Science & Engineering, Ritsumeikan University

4. 重み付け K 近傍法による相手の行動予測

本章では、重み付け K 近傍法による相手の行動予測について述べる．このアルゴリズムをデータの収集、行動決定の 2 つに分けて述べていく．

4.1 データの収集

相手が攻撃行動を行うときの、攻撃の種類と相対座標を収集する．1 試合を通して相手が攻撃を行うたびにデータを収集し続けていき、収集したデータを座標空間上にプロットしていく．

4.2 行動予測

行動を決定する前に、まず相手が攻撃を行うかどを判断する．現在の相対座標を取得し座標空間上にプロットする．その座標から一定距離内に上記で収集したデータが一定数以上あれば攻撃を行うと判断する．攻撃判定に用いるこれらの距離、数の値は事前に任意の値を設定しておく．

相手が攻撃を行うと判断した場合、相手の攻撃を予測する．まず座標空間上において現在の相対座標の近傍に存在する K 個のデータを参照する．参照したデータの中で攻撃の種類ごとに、現在の相対座標との距離の逆数の平均値を重みとした値を以下の式(2)を用いて求める。

$$\frac{1}{N} * \sum_{n=1}^{N} \frac{1}{D_n} \qquad (2)$$

N は近傍に存在する同じ攻撃データを保持するデータ数である。D_nは各データの現在の相対座標に対するユークリッド距離である。上記の式(2)で求めた値が最も大きい攻撃行動を相手が行うだろうと予測する．

5. 実験と結果

5.1 実験方法

提案手法の動的難易度調整を実装することにより、ユーザスキルに応じて AI の強さを調整することができているかを実験によって検証する．検証を行うために、動的難易度調整を実装した AI と実装していない AI との対戦スコアを比較する．動的難易度調整を実装していない AI は常に評価値が最大となる行動を行う AI である。また、今回の実験では τ の値を 500 に設定した．K 近傍法で用いる K の値は 5 に固定して実験を行う．

実験は上記の 2 種類の AI に対して対人戦を行う。10 人に各 AI と対戦を行ってもらい、スコアを比較する。実験を始める前にゲームルール、操作方法を説明し、練習として練習用の AI と 2 試合対戦を行う。練習用の AI はランダムに行動する AI である。練習後に上記の実験用の 2 種類の AI と各 5 試合ずつ対戦を行う。また、被験者を 5 人ずつ 2 つのグループⅠ,Ⅱに分けておき、グループⅠは動的難易度調整を実装していない AI から先に 5 試合行い、次に動的難易度調整を実装した AI と対戦を行う。グループⅡはその逆の順番で試合を行う。この際に被験者には AI の概要は伝えずに対戦を行う。なお、今回の実験では、格闘ゲーム初体験の人から慣れている人までからまんべんなく被験者を 10 人選出した。

5.2 実験結果

表 1,3 に両グループの動的難易度調整を実装していない AI と対戦した際の被験者側が獲得した各ラウンドの平均スコアと合計平均スコアを示す。被験者 J を除き、両グループともに被験者側のスコアが低いことが分かる。これは AI が K 近傍法を用いて相手の行動を予測し、有利な行動を選択しているためである．表 2,4 に動的難易度調整を実装した AI と対戦した際の被験者側の獲得平均スコアを示す。動的難易度調整を実装していない AI との対戦結果と比較すると、被験者 J を除いた被験者側のスコアが上昇し、500 に近づき、AI のスコアが低く抑えられていることが分かる．これは τ を設定したことにより、AI の行動決定の方法が変化したためである．

しかし被験者 J に関しては、動的難易度調整を実装していない AI に対して高いスコアを残している。これは被験者 J のユーザスキルが AI の強さを上回っていたため、AI 自身の強さを抑えていくことによる動的難易度調整を行うことができなかったためである。

表 1　グループⅠ 動的難易度調整を実装していない AI との対戦結果

τ なし	Round1	Round2	Round3	Game
被験者A	540	325	333	399
被験者B	366	234	116	239
被験者C	204	183	143	177
被験者D	106	131	64	100
被験者E	282	240	228	250

表 2　グループⅠ 動的難易度調整を実装した AI との対戦結果

τ =500	Round1	Round2	Round3	Game
被験者A	575	467	554	532
被験者B	466	480	430	459
被験者C	405	349	443	399
被験者D	512	417	576	502
被験者E	480	373	421	425

表 3　グループⅡ 動的難易度調整を実装していない AI との対戦結果

τ なし	Round1	Round2	Round3	Game
被験者F	335	271	240	282
被験者G	341	253	254	282
被験者H	216	196	139	184
被験者I	290	237	197	241
被験者J	633	407	502	514

表 4　グループⅡ 動的難易度調整を実装した AI との対戦結果

τ =500	Round1	Round2	Round3	Game
被験者F	470	477	328	425
被験者G	398	449	498	448
被験者H	454	494	519	489
被験者I	315	455	420	397
被験者J	502	464	426	464

6. おわりに

本稿では、格闘ゲームにおける動的な難易度調整としてスコアに応じて AI の行動選択を変化させる手法を提案した。あらかじめ τ を設定することにより、動的な難易度調整を実現することができている。しかし、被験者 J のように高度なユーザスキルを持つ相手にも対応できるように、より強い AI を実現する必要がある。今後の研究では、パラメータ K を動的に変更することにより、より正確に相手の行動予測を行うことで強い AI の実現を目指す。

参考文献

[1] Jenova Chen: “Flow in Game,” Communications of the ACM, Vol. 50, No. 4, pp. 31-34, 2007.

[2] B.H. Cho, S.H. Jung, Y.R. Seong, and H.R. Oh, “Exploiting Intelligence in Fighting Action Games using Neural Networks,”IEICE Transactions on Information and Systems, vol. E89-D, no. 3, pp. 1249–1256, 2006.

[3] S, Lueangrueangroj and V. Kotrajaras, “Real-time Imitation based Learning for Commercial Fighting Games,” Proc. of Computer Games,Multimedia and Allied Technology 2009, pp. 1–3, 2009.