Can Opponent Models Aid Poker Player Evolution?

R.J.S.Baker, *Member, IEEE*, P.I.Cowling, *Member, IEEE*, T.W.G.Randall, *Member, IEEE*, and P.Jiang, *Member, IEEE*,

Abstract—**We investigate the impact of Bayesian opponent modeling upon the evolution of a player for a simplified poker game. Through the evolution of Artificial Neural Networks using NEAT we create and compare players both utilizing and ignoring Bayesian opponent beliefs. We test the effectiveness of this model against various collections of dynamic and partially randomized opponents and find that using a Bayesian opponent model enhances our AI players even when dealing with a previously unseen collection of players. We further utilize the inherent recurrency of our evolved players in order to recognize the opponent models of multiple players. Through ablative studies upon the inputs of the network, we show that utilization of an opponent model as an evolutionary aid yields significantly stronger players in this case.**

I. Introduction

Poker has simple rules, with many layers of tactical complexity [1]; each player is dealt a set number of cards (which varies for different types of poker, and is generally between one and seven), and then one or more betting round(s) take place, within which each player tries to convince their opponent that they have the best hand. The layers of complexity appear in the performance of the betting round, as each player's betting action can represent a strong hand or a bluff. Bravado through bluffing can either be successful or ruinous for a player, and knowing the right time to play a hand separates successful players from mediocre ones. The three actions that can be performed in-game are common to all forms of poker, as follows:

- ***Bet/Raise***: Add money to the pot, and increase the monetary risk for the bettor and the opponents.
- ***Check/Call***: Make the smallest bet required to stay in the hand (which may be nothing).
- ***Fold***: Take no further part in the proceedings of the hand.

[1]

These basic actions are an essential staple of all poker games. It is the underlying strategy behind the decision making process of a player that makes the game of poker arguably one of the most skilful card games in the world. The complexity of Poker play results largely from the fact that the only information available to a player of the game's state is that of their card(s) held, any community cards known to all, and that of any past actions the opponents have made. This paper investigates the analysis of those past actions using Bayesian inference in order to use the resulting probabilistic model to aid the evolution of an Artificial Neural Network (ANN) controlled agent. Specifically we assess the performance of agents controlled by evolutionary neural networks, with and without access to Bayesian opponent models, in a number of scenarios. We investigate a single-card simplification of poker that nevertheless captures some of the important tactical subtleties of the full game, while being more amenable to analysis.

Poker has driven numerous research efforts for many years, early efforts including Findler's research into machine cognition using Poker, judging that dynamic or opponent-adaptive play is necessary in order to be successful, and that static play styles can be easily beaten once the style has been learnt [2]. However, Poker has been overshadowed somewhat in comparison to games such as Chess [3]. One of the reasons for this is arguably due to the difficulty (and combinatorial explosion) that results from having imperfect information.

Even though it is intrinsically possible to generate game theoretically optimal strategies for poker [4], this analysis would take far too long to conduct in practice, even for the simple version of poker which we consider in this paper, partly due to the prospective use of bluffing: an opponent's bet could represent a bluffing move, or a show of confidence, this being difficult for any player to determine. Several approaches to understanding the mechanics behind games of imperfect information have been based upon simplified variants of poker [4], [5], [6]. In recent years, great strides in creating an artificially intelligent Poker player for full poker have come from Darse Billings and University of Alberta's GAMES group [7], [8], [9]. Billings reduces the complexity of the gaming situation by eliminating betting rounds; simplifying the problem, but maintaining the fundamental nature of Poker. Billings' approach to opponent modeling uses a predictive neural network based system that, when given a set of inputs of an opponent's last action and the current state of play, will produce a probability distribution of the opponent's next action. Our approach is such that instead of simply predicting an opponent's next action, we predict the opponent's playing style in relation to past actions performed, against a small set of possible styles. Schaeffer [9] defined some 'ground rules' for creating a world-class poker player, forming an integral part of the *Loki* system. These requirements include *hand strength*, *betting strategy*, *bluffing*, *unpredictability* and o*pponent modeling*. This paper will compare players with and without opponent models to investigate this requirement.

Opponent modeling has been seen as having a greater impact on success in games of poker than most other games,

R.J.S.Baker, P.I.Cowling, T.W.G.Randall and P.Jiang are with the MOSAIC Research Centre, Department of Computing, University of Bradford, BD7 1DP, UK. E-mail: R.J.S.Baker, P.I.Cowling, T.W.G.Randall and P.Jiang@bradford.ac.uk

indeed poker is an important testbed for opponent modeling research. In the case of [10], modeling is implemented by adjusting weights representing beliefs in an opponent's cards. Similarly, Saund's approach captures and analyses betting actions in relation to inferring the downcards held by an opponent in seven-card stud poker, as well as determining that opponent action analysis is important in determining successful play [11]. Barone and While's poker work [12], [13], [14] has involved the investigation of evolutionary approaches to play against various 'styles' of opponent. Our previous work [15] involved opponents of a similar nature to Barone's, and described a means of representing them, which we will use again in this paper, in order to determine appropriate reactions.

II. One Card Poker

Our research uses a simple version of poker, as defined in our previous work, which still maintains useful tactical ideas from full-scale poker [15]. The deck consists of ten cards, numbered 1, 2, .., 10. Each player has an initial credit of 10 chips, and each hand entered requires a one-chip ante from each player, after which each player is dealt one card. This approach is similar to that of Koller and Pfeffer [4], which uses an 8-card deck to find an optimal mixed strategy using game theory with each player having only one card and one chip each, and Burns [16], which investigates the optimality of commonsense poker strategies, uses a deck in which each player is dealt a card classed as 'high' or 'low'. The winner of the hand is the player with the highest valued card at the showdown, or the last player left if all opponents fold.

After the cards are dealt, players make a decision whether to fold, check or bet given the value of their card. Betting (which is equivalent to a 'raise' action), and each subsequent reraise costs one chip. Once all players have matched one another's bets (or all but one player has folded) the showdown is reached, and the player with the highest card (or only player remaining) receives the pot. The players continue playing further hands until there exists a tournament winner who has won all of the chips. In our previous work [15], there existed no betting limit, which culminated in some tournaments ending after a single hand. In this current work, a bet limit of 4 chips per player per hand is employed. The strategies which can be employed in this version of Poker, particularly of bluffing and opponent modeling, echo those that can be employed in a full-scale Poker game.

III. The AI Players

A. Distinct Style Players

Poker players may usefully be categorized into four main styles:

- ***Loose Aggressive (LA)***: A player that typically over-values hand strength, who will constantly force the pot higher, even with a relatively weak hand.
- ***Loose Passive (LP)***: A player that will also over value their hand, but will generally call, and only bet when they believe that they are likely to win the hand.
- ***Tight Aggressive (TA)***: A player that accurately values their card, and will fold more often, but any hand where a high card is held, then the player will bet aggressively.
- ***Tight Passive (TP)***: A player that plays very few hands, and even when doing so will generally call, and only bet in rare situations when a win is most likely.

Barone and While recognized these play styles as part of their investigation into evolutionary adaptive poker play, and have also been utilized as part of Kendall and Willdig's work [12], [13], [14], [17].

Each of these styles of player was created using a simple deterministic design [Fig. 1]. A player's style is characterized by a probability pair (α, β), where α represents the minimum win probability (the probability this player has the best hand) required for a player to remain in the hand, and β represents the minimum win probability for the player to bet. Then α is responsible for whether a player is tight or loose, and β for whether a player is passive or aggressive. If the win probability is less than α, the player will make a checking action if no money needs to be placed in the pot to remain in the hand, and fold otherwise. It should be noted that these players act on card strength alone, and ignore opponents' actions.

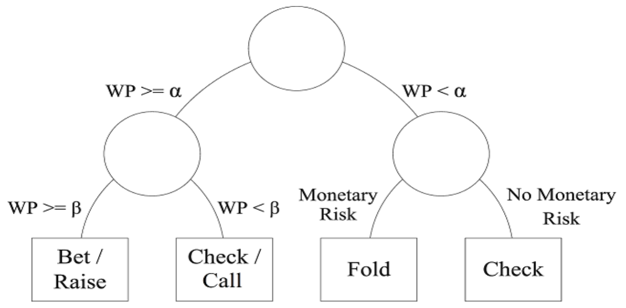

Fig. 1. Layout of a simple player

A pair (α, β) represents a deterministic player, with a distinct play style. The α and β values for each playing style are defined in Table I.

TABLE I
α AND β VALUES FOR EACH STYLE OF DETERMINISTIC PLAYER

	A	β
LA	0.1	0.2
LP	0.1	0.9
TA	0.5	0.6
TP	0.5	0.9

B. Evolving an ANN-Controlled Opponent

In our previous work, "Anti-Players" were created as a 'nemesis' to each of the LA, LP, TA, and TP players. (α, β) pairs were tested in 0.1 increments for $0 \le \alpha \le \beta \le 1$ to determine the best (α', β') pair against each opponent style.

Although our previous agent's approach involved a dynamic means of learning to approach an opponent, the strictly static nature of the agent's response renders it unable to use more complex tactics such as check-raises, for example. Our motivation in this paper is to investigate the potential for evolving a player to be able to develop complex strategic behaviours. In this investigation, we evolve a player using a C#.NET implementation of NEAT, SharpNEAT [18]. Our reasoning behind using the NEAT algorithm is due to the successes of NEAT's topological and weight evolution in finding a suitable network structure for various problems, including game-playing agent control and pole-balancing experiments [19], [20].

An issue for every researcher using ANN's in their work is that of the number and form of their inputs, Davidson [21], [22] used ANN's to predict an opponent's next action, and mainly used binary values for Boolean inputs representing the stage of the game and the last action of an opponent, and real values from 0 to 1 for all others (such as pot odds) in order to represent opponent playing habits. The design of our network structure (which is simpler than Davidson's approach due to the format of our game) can be seen in Fig. 2.

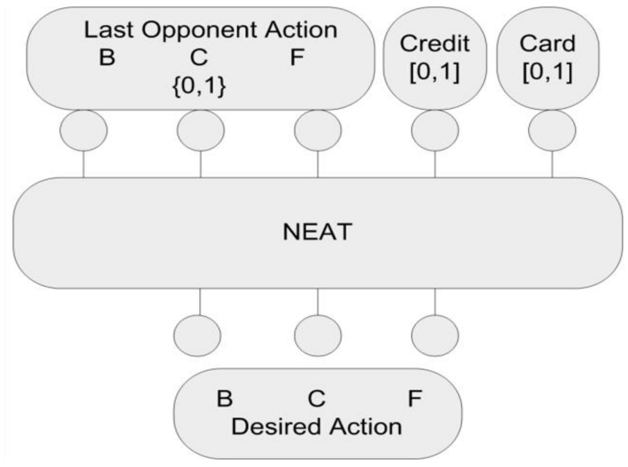

Fig. 2. Design of player network

The 'Last Opponent Action' inputs translate the last action of the opponent into binary. The 'Current Credit' input represents the ratio of chips the player has in comparison to the number of chips available at the table (including all opponent-held chips). The final (and arguably most important) input is that of the card held, which is represented by its number divided by the total number of cards. The outputs of the network represent the action to be taken, the action represented by the largest numerical output is the action performed.

IV. Experimental Results

The evolution of an ANN-controlled player can be seen in Figs 3 through 6. All experiments are run on a Pentium Core 2 Quad 2.4 GHz with 4GB RAM using C#.NET 3.5 running under Windows Vista 64-bit SP1. In these experiments the fitness of each of the genomes in a population is represented by *T*, the percentage of tournaments won by the player (i.e. the tournaments where this player wins all the chips of all opponents) over 100 tournaments. We use a population size of 100, a node-addition probability of 0.005, and a node-connection addition probability of 0.01. The node addition probability represents the probability that a new node will be added to the network, and the node addition probability represents the probability of adding a new connection between any two nodes of the network. These are indicative of the mutative stage of the evolution. There are further capabilities within NEAT to *destroy* connections and nodes, but have not been enabled here, as unsuitable solutions are generally evolved out of the genome, lessening the need for such destructive measures on potentially promising solutions. Each of the subsequent graphs has been averaged over 5 individiual runs; note that the 'best fitness' in these graphs represents the average best fitness, and that these results have been run for 400 generations, but graphs have been pruned in order for ease of reading, and has been done so only when no further improvement has been exhibited by the evolution. It should be further noted that, with respect to the 'best' solution at the end of evolution, the results given against LA and LP opponents have a standard deviation of 2 tournament wins (2% of all tournaments played in this case), and those against TA and TP opponents are accurate to within 5 tournament wins (5%). This is due to the stochastic nature of the cards in determining success, as well as the difficulty of playing against tighter styles of player.

As we can see (Figs 3-5), network evolution versus a single static opponent (otherwise known as 'heads-up' play) works sufficiently well, with opposition to LA, LP, and TA styles reaching a 'best' success rate of 100%, with average population fitness above 90%. This is unsurprising, as it has been seen that even a static counter-approach to defeating static styles can be successful [2], [15], and here we use a dynamically learned approach.

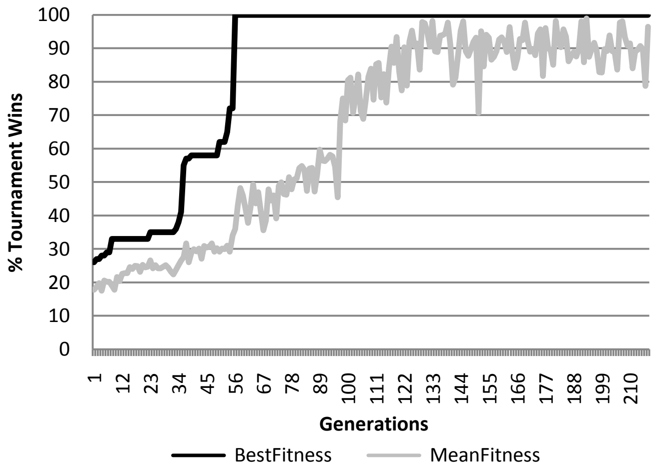

Fig. 3. Evolution of an ANN against a simple Loose Aggressive Player

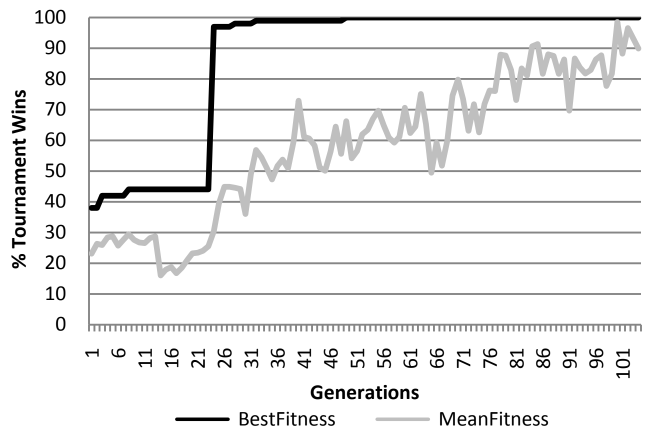

Fig. 4. Evolution of an ANN against a simple Loose Passive Player

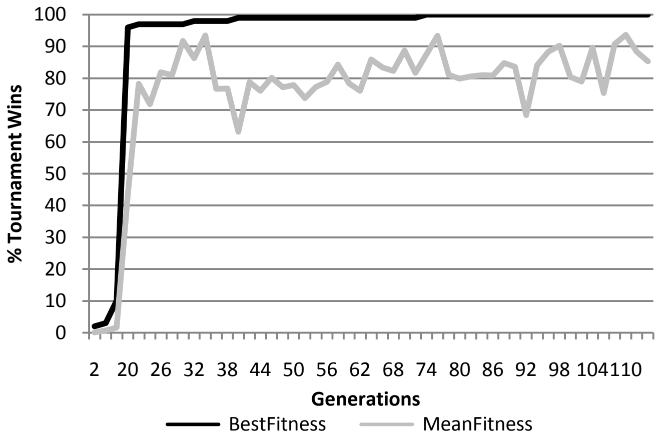

Fig. 5. Evolution of an ANN against a simple Tight Aggressive Player

When we observe Fig. 6, however, we notice that our evolution fails to reach an exceptionally high success rate against a single Tight Passive player, even after many generations. This is due to the Tight Passive strategy being stronger than the other static strategies, such that our previous work [15] labeled them as the most 'threatening' play style, and also the hardest to gauge due to the frequency at which checking actions occur over any display of strength. Nonetheless, even in this case tournament success of the evolved NEAT network is impressive.

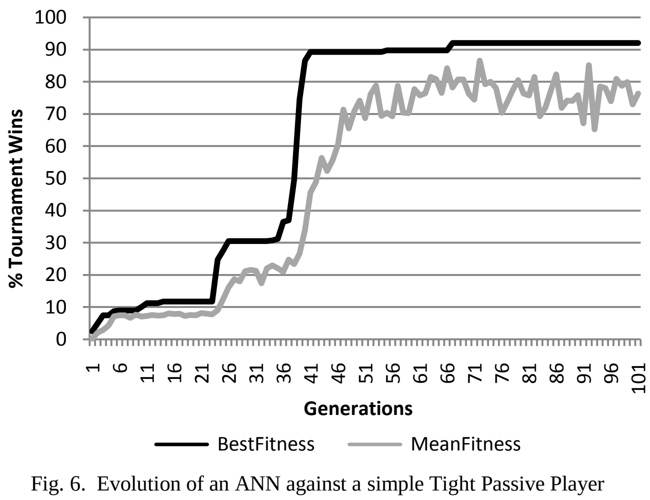

Fig. 6. Evolution of an ANN against a simple Tight Passive Player

Given that a NEAT-evolved player is capable of defeating a single individual style, we aim to evolve a player capable of defeating all four types of opponent in order to reduce the necessity of selection between individual strategies; Fig. 7 shows the evolution of a player continuing to use the inputs described in Fig. 2. We make each candidate solution play 100 games against each opponent in every generation, such that they will play 100 games against an LA opponent, then 100 games against an LP opponent, and so on. The fitness *f* of each solution is again the percentage of tournaments won against all opponent styles.

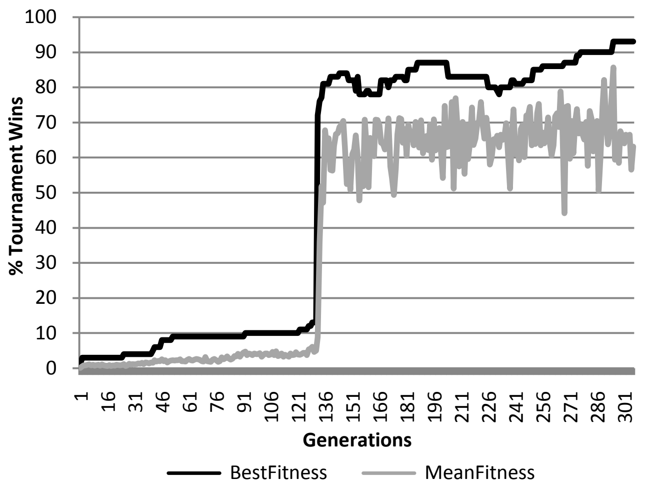

Fig 7. Evolution of an ANN against all types of opponent.

A. *Augmenting Evolution with Bayesian Analysis*

It is noted that the evolved player average reaches a maximum average success rate of 87% over 100 tournaments. If we consider Figs. 2-5, we can tell that our players should potentially be able to gain a greater success rate than this against these opponents. An opponent model can aid in representing the individual nature of an adversary, and as such could aid in the correct selection of an appropriate reaction to each opponent. Fig. 8 shows the structure of our proposed network design.

Our previous work [15] has emphasized that Bayes' rule can be a powerful learning approach that can analyse the past play information of an opponent, and determine useful information about each opponent's respective playing style.

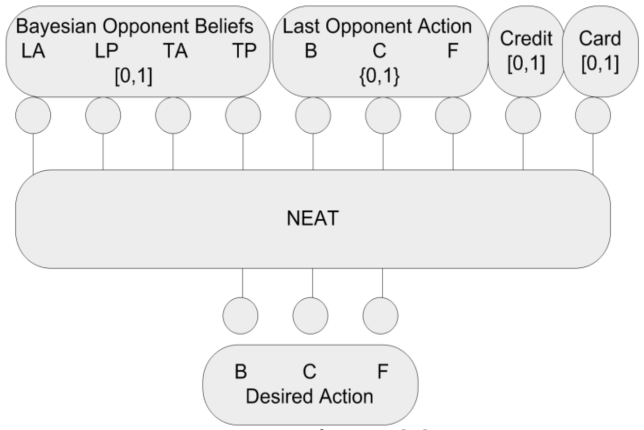

Fig. 8. Design of proposed player

The usage of Bayesian probabilities to model uncertainties is popular in relation to imperfect information games such as Poker [16].

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)} = \frac{P(B \mid A)P(A)}{\sum_a P(B \mid a)P(a)} \quad (1)$$

Equation (1) gives Bayes' rule where A is a random variable representing player type, and B is a random variable representing player action. Our calculation uses an a priori belief of 0.25 as P(A) for each of the four player strategies for the first iteration. The probabilities in Table III represent an *a priori* P(B | A) which were evaluated by analysing the past actions of players of style A over 100,000 games. P(A) is the prior belief of an opponent's play style, and as such is set to 0.25 initially as all opponent styles are assumed equally likely at the start of a game. Bayes' rule updates the initial probability of player style belief such that P(A | B) for the current iteration becomes P(A) for the next action analysed (i.e. our belief of style P(A) is the result of P(A | B) for the last action analysed). P(B) is represented by the summation of P(B | A)P(A) for all possible A (represented by a), and is used as a normalising constant.

TABLE III
ACTION PROBABILITIES FOR EACH OPPONENT STYLE

	FOLD	CHECK /CALL	BET /RAISE
LA	0.36	0.05	0.59
LP	0.60	0.29	0.11
TA	0.73	0.02	0.25
TP	0.87	0.07	0.06

These per-style beliefs are used as four further inputs to augment the evolution of our player. Fig. 9 displays the effect that opponent model augmentation has upon the evolution.

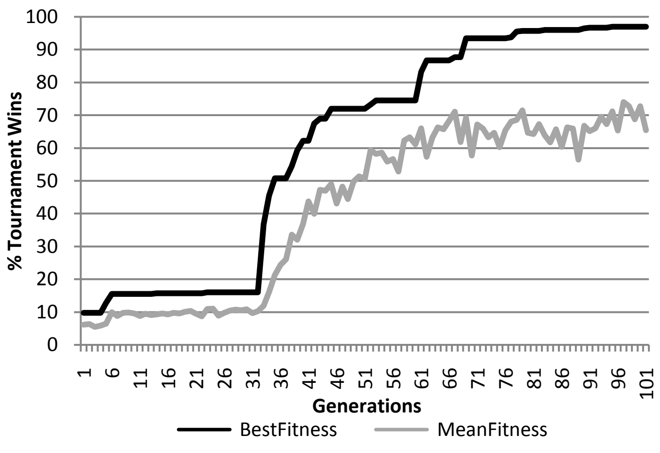

Fig. 9. Evolution of a Bayesian Model-augmented ANN against all opponent styles.

We can see that the network quickly evolves to a solution which reaches a best tournament success of an impressive 97% over 100 tournaments. The limitation of average population performance can most likely be attributed to the increased number of inputs, which increases the search space and increases the difficulty of learning. In the 400 generations of evolution, no further improvement was seen after the first 100.

B. Increasing the Complexity

Much research into Poker has looked into simple two-player, one-on-one games (more commonly known as 'heads up' Poker) [4], [5], [7], [12], [16], [23], which reduces evaluation complexity, especially in relation to dealing with opponent models. The increased complexity of a 3 vs. 1 game, for example, calls for our neural network to interpret the information of 3 opponents instead of 1 [6], [11], [15]. We believe that the ability of NEAT to evolve network topologies as well as weights might lend itself to such a problem, particularly due to the strong possibility of creating *recurrent* neural networks. Our aim of reacting to more than one opponent shall take advantage of the 'memory' afforded to us through the recurrent connections [24]. We use the same network inputs as Fig. 8, and iteratively pass the inputs for the first second, third, and (potentially) nth opponent. If evolution allows, this will result in a 'memory' of the previous opponents which should influence the current decision. After the final opponent's data is input, the output is received, and the action represented by the largest numerical output is performed (through analysis of the outputs we have seen that, after evolution, these values usually give a very clear decision). The fitness f for this experiment is represented by Equation (2), where a represents the number of player styles, T represents the fraction of tournaments won (over 100 tournaments), H_w represents the number of hands won by the player, and H_p represents the total number of hands played.

$$f = (\textstyle\sum_a T)a^{-1} + \frac{H_w}{H_p} \quad (2)$$

The ratio of hand success is added in order to aid initial evolutionary candidates due to the increased complexity of evolving a player against three opponents. Initial results from evolution without this value had difficulty in evolving initial reasonable strategies.

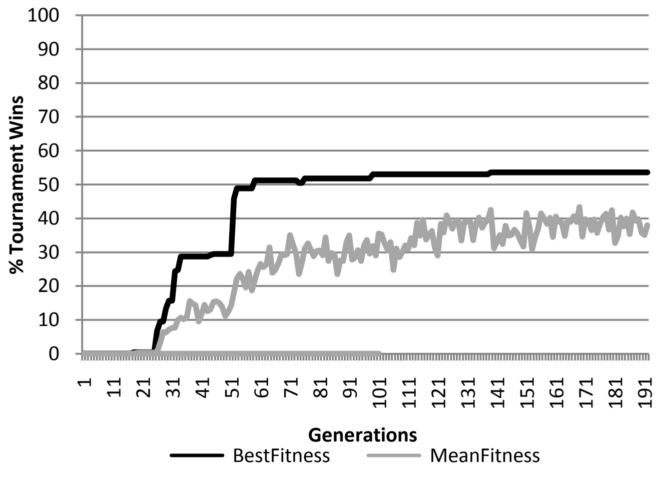

Fig 10. Evolution of a three-input recurrent network against all four types of opponent in a four-player scenario

As can be observed in Fig. 10, the success of the player (without an opponent model) is modest with a best tournament win percentage (over 100 tournaments) of 54%, and an average win percentage of 38%, higher than the 25% which a set of 4 equally-matched players would obtain. When we use Bayesian inference of opponent actions (Fig. 11), however, we obtain an average success rate of over 70%, the same as that enjoyed by the same approach in the simpler 1 vs. 1 environment from Fig. 9, and a best tournament success rate of 97%.

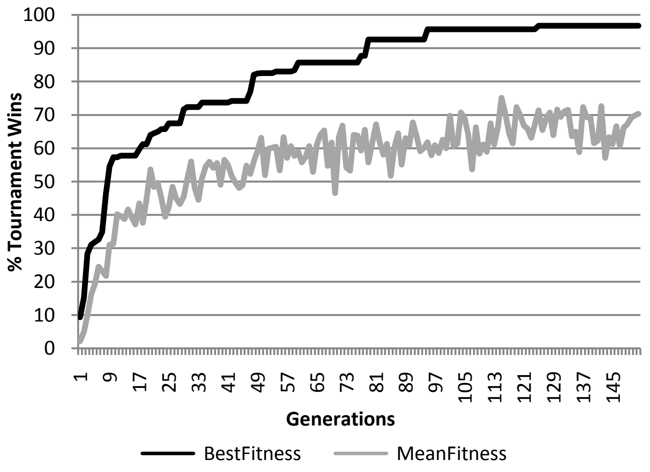

Fig 11. Evolution of a Bayesian-augmented ANN against all types of opponent in a four-player scenario

The fitness of our evolved players was measured purely against tables of matching types; one table against three LA players, one against three LP players, one against three TA players, and one against three TP players. In Fig. 12 we illustrate the tournament success of the best solution from Fig. 11's evolution against all possible mixed player-table combinations, and as we can see, the network has excellent success rates against opponent combinations it *was not trained against*. This is arguably due to the advantage a player receives once it has access to beliefs about an opponent's style of play. An issue of note with Fig. 12 is the low success rates against tables of primarily loose players, including tables which consist of three LA, and three LP types, these being one of the four combinations it was *evolved against*. The nature of a 4-player game of poker differs from a 1 vs. 1 game, as a greater number of loose opponents mean that there is a greater probability of an opponent holding a good hand than if there was only one adversary. The choice of action in this case is therefore made much harder given the loose nature of all the opponents.

The inclusion of an opponent model appears to aid both the evolution and decision process of the agent. This appears to be due to the ANN's ability to utilise the opponent model's separation of opponent types in order to determine a reasonable course of actions against the opponent(s). In order to test this theory, we perform an ablative study upon our evolved 'best' network. For this, we disable sets of inputs to our network to observe how well the player can cope without certain abilities available. In this experiment we compare three inputs that we feel are essential to the players' function, namely opponent model information, last action information, and the recurrent nature of our evolved networks.

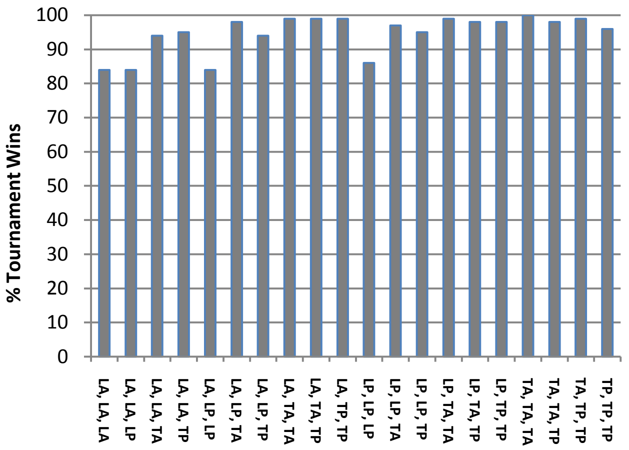

Fig 12. Tournament performance of the best evolved genome against all combinations of 3 opponents over 100 tournaments

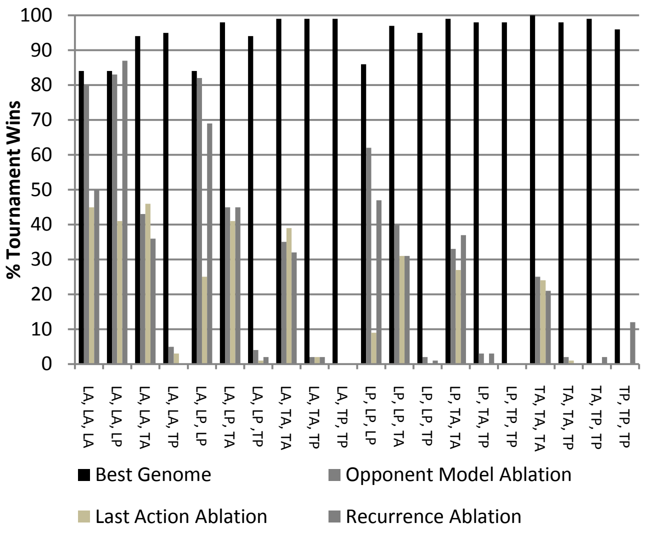

Fig 13. Tournament performance of ablated configurations of the 'best' evolved genome over 100 tournaments

Fig. 13 shows the difference in the fraction of tournaments won when each of the inputs of the player's network are disabled. The greatest difference appears in relation to the removal of recurrence (in this case, only a single player's data is passed to the network, excluding all players at the table) and that of the most recent action by the opponent. The removal of recurrence removes the iteratively-passed opponent data, and hence memory of opponent characteristics, which greatly impairs performance. A noticeable facet of this is related to tables of three similarly-typed opponents; in Fig. 13 we can see that in each of the situations where there are three opponents of the same type, the loss of recurrence causes a slightly less damaging effect. The removal of most recent action has a significantly damaging effect upon the success of our player, which is understandable due to the importance of taking our opponent's most recent action to infer the strength of their downcard. As for the removal of opponent model, the effect is again pronounced in all cases (although less so than the other two effects above). We believe that this means that the opponent model is integral to the operation of our neural network in terms of being able to determine a strategy tailored to the nature of the opponents.

C. Dynamic opponents

In order to test the ability of our player to adjust to the potentially dynamic nature of opponents, we create two sets of tests: firstly we test our player against each of the standard four types of opponent, but we make our opponent bluff (bet when the player decides it should actually fold) with probability *p*, which is adjusted in increments of 0.01where $0 < p < 1$. Fig 14 shows the results of these tests.

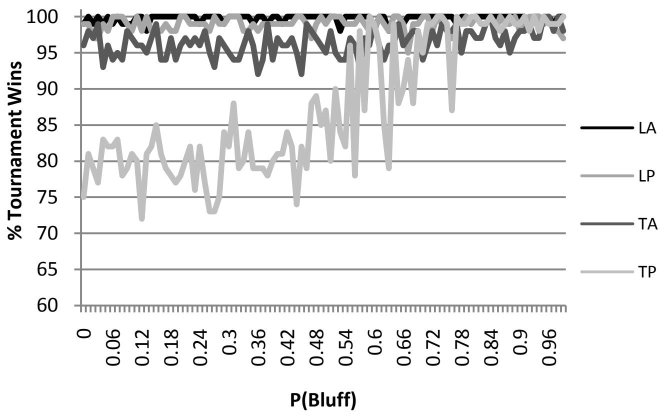

Fig 14. Tournament performance of the 'best' evolved genome against partially randomized opponents over 100 tournaments

"Bluffing" improves the performance of a TP player: at a bluff probability of 0.12 our player drops to a 72% tournament success rate. This is understandable, as approaches call for a level of bluffing in order to attain an optimal strategy [4], [5]. As the chance of performing a bluffing action rises, so does the success of our player against the bluffing TP player. Reasoning for this is due to the nature of player styles, our player's model now assumes that the opponent is loose aggressive, and caters for the eventuality that our randomized player will be bluffing a large portion of the time. As for the other types of opponent, they will all mostly be classified as being an LA player as *p* rises. Bluffing does not improve play quality of LA, LP or TA players.

If a players' current tactic is unsuccessful, then it is sensible for the player to change their current strategy; we implement a series of players that transition from one style to another once their chips are at a level that is 50% of their initial chips at the start of the game. We can see in Fig. 15 that our player is able to cope suitably against our varied-style opponents. It is notable that our player is most susceptible to opponent strategy change when moving from a weaker style to a tight one. The main reasoning behind this is that our Bayesian modeler has to readjust its style-representative weights in order to accommodate the opponent's shift in style, and as such a lag is involved in "understanding" the opponents' strategy. The transition from a stronger style to a looser one is not likely to be an effective strategy, and our results confirm this. However, moving from a tight style to a loose one can be a good way of scaring opponents out of the game until they realize the strategy in play. The failing of this approach against our ANN strategy, however is that the probabilistic way in which our modeler updates its beliefs means that these beliefs will be altered significantly when a tight player repeatedly performs an action that it should rarely do (the main example being to move from TP to LA, drastically increasing the frequency of betting actions; see Table III).

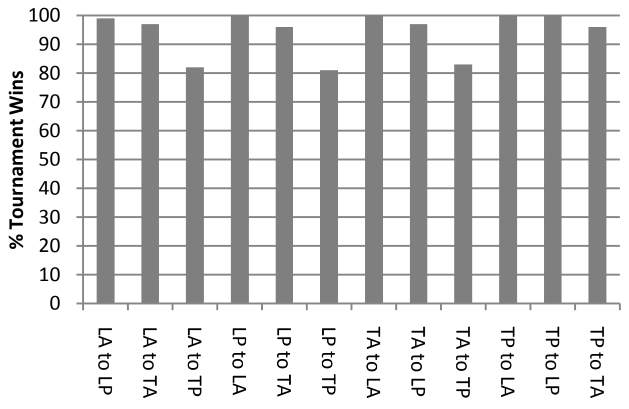

Fig 15. Tournament performance of the 'best' evolved genome against dynamically styled opponents over 100 tournaments

V. Conclusion

In this paper players are evolved using NEAT for a simplified game of poker. We first show it is easy to evolve a player against individual opponents of a fixed style. We then compare the utility of opponent models in aiding the evolution and performance of game-playing agents against players that do not make use of such information. Against a single adversary, the results show little difference between approaches that use an opponent model, and those which do not. We then compare our players in an environment where there is more than one opponent. Results show the benefits of using opponent models increase with increased numbers of opponents. In this instance, the evolved player relies upon Bayesian opponent modelling and the recurrent nature of the evolved neural network is shown to be crucial in order to apply the appropriate strategy for each set of different opponents, and is able to generalise and defeat tables of opponent combinations not yet encountered. Furthermore, we test the approach against opponents that employ simple bluffing tactics, as well as simple dynamic strategy approaches. In these experiments we find that our opponent-model augmented NEAT networks are able to perform well against these dynamic opponents.

Through our own (human-computer) interactions with the 'best' evolved player, we have noted that it plays *tightly*, which is good for a close game, but does not take advantage of any potential for bluffing, as well as being very susceptible to bluffing by the opponent.

VI. Future Work

In future, we aim to investigate further methods for representation of a game player in terms of their tactical strengths and weaknesses. Playing effectively against an opponent requires the *discovery* of the opponents' weaknesses. Our aim is for a player to discover these tactics for itself before using a means of action selection tailored to the opponent. Our future work aims to evolve a bluff-aware approach; this would include finding a means of discovering when our opponent is bluffing, and to augment our AI players so that they can successfully perform deceptive strategies. As for our current approach, we feel that using our opponent model could potentially be upscaled to evolving an agent for full-scale Texas Hold 'em, but some simplification may need to arise in relation to betting rounds, as well as the representation of cards held and hand strength, as well as the potential for cards to improve, or conversely worsen. The number of ANN inputs would potentially need to drastically increase in order to evolve a decision-making agent, but we shall investigate the potential for improving the evolution of a more complex agent using opponent models.

References

[1] D. Sklansky, *The Theory of Poker*. Two Plus Two Publishing, 1992.

[2] N. Findler, "Studies in Machine Cognition Using the Game of Poker." *CACM* 20(4), pp 230-245, 1977

[3] M. Campbell, A.J. Haone Jr, F-h. Hsu, Deep Blue, *Artificial Intelligence,* 2002, Vol.134, (pp.57-83)

[4] D. Koller, A. Pfeffer, Representations and solutions for game theoretic problems *Artificial Intelligence*, 1997, (pp.167–215)

[5] J. von Neumann, O. Morgenstern, *Theory of Games and Economic Behavior*. Princeton NJ: Princeton Univ. Press, 1944

[6] M. Sakaguchi, S. Sakai, Solutions of some three-person stud and draw poker. *Mathematics Japonica* 1992, (pp. 1147-1160)

[7] D. Billings, N. Burch, A. Davidson, R. Holte, J. Schaeffer, T. Schauenberg, and D. Szafron, Approximating game-theoretic optimal strategies for full-scale poker *In Proceedings of the eighteenth International Joint Conference on Artificial Intelligence* 2003, (pp. 661-668).

[8] D. Billings, A. Davidson, J. Schaeffer, and D. Szafron, The challenge of poker. *Artificial Intelligence Journal,* 2002, (pp. 201–240).

[9] J. Schaeffer, D. Billings, L. Peña, D. Szafron, Learning to play strong poker *In ICML-99, Proceedings of the 16th International Conference on Machine Learning,* 1999

[10] D. Billings, D. Papp, J. Schaeffer, D. Szafron, Opponent modeling in poker *Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence,* 1998, (pp. 493 - 499)

[11] E. Saund, Capturing the information conveyed by opponents' betting behaviour in poker. *In proceedings of 2006 IEEE Symposium on Computational Intelligence and Games (CIG),* (pp. 126-133)

[12] L. Barone, L. While, An adaptive learning model for simplified poker using evolutionary algorithms *In proceedings of Congress of Evolutionary Computation 1999* (CEC'99), July 6-9, Washington DC, (pp 153-160).

[13] L. Barone, L. While, Evolving Adaptive Play for Simplified *Poker. In proceedings of IEE International Conference on Com*putational Intelligence (ICEC-98), pp 108-113, 1998.

[14] L. Barone, While, L. Adaptive Learning for Poker. *In proceedings of the Genetic and Evolutionary Computation Conference*, pp 566-573, 2000.

[15] R.J.S. Baker, and P.I. Cowling, Bayesian Opponent Modeling in a Simple Poker Environment, *IEEE Symposium on Computational Intelligence and Games (CIG 2007)*, Honolulu, USA.

[16] K. Burns, Style in poker, *In Proceedings of 2006 IEEE Symposium on Computational Intelligence and Games (CIG),* (pp.257-264)

[17] G. Kendall and M. Willdig, An Investigation of an Adaptive Poker Player. *In proceedings of 14th Australian Joint Conference on Artificial Intelligence*, 2001, pp. 189-200.

[18] D.B. D'Ambrosio, K.O. Stanley, A novel generative encoding for exploiting neural network sensor and output geometry, in *Proceedings of the Genetic and Evolutionary Computation Conference*

[19] K.O. Stanley, R. Miikkulainen, Evolving Neural Networks through Augmenting Topologies. *Evolutionary Computation* 2002, 10 (2): 99-127

[20] S. Whiteson, P. Stone, K.O. Stanley, R. Miikkulainen, N. Kohl, Automatic Feature Selection via Neuroevolution. *In Proceedings of the Genetic and Evolutionary Computation Conference,* 2005.

[21] A. Davidson, (1999) Using Artificial Neural Networks to Model Opponents in Texas Hold 'Em. [Unpublished manuscript]. Available: http://spaz.ca/aaron/poker/nnpoker.pdf.

[22] A. Davidson, D. Billings, J. Schaeffer, and D. Szafron, Improved Opponent Modeling in Poker. *Proceedings of the 2000 International Conference on Artificial Intelligence (ICAI'2000).* 1999, 1467--1473.

[23] F. Southey, M.P. Bowling, B. Larson, C. Piccione, N. Burch, D. Billings, D. Rayner, Bayes' Bluff: Opponent Modelling in Poker. *In Proceedings of the 21st Annual Conference on Uncertainty in Artificial Intelligence* (UAI-05), 2005, pp 550-555

[24] J.L. Elman, Finding structure in time. *Cognitive Science*, 1990, 14, 179-211.