Evolving Adaptive LSTM Poker Players for Effective Opponent Exploitation

Xun Li and Risto Miikkulainen
University of Texas at Austin
xun.bhsfer@utexas.edu | risto@cs.utexas.edu

Abstract

In many imperfect information games, the ability to exploit the opponent is crucial for achieving high performance. For instance, skilled poker players usually capitalize on various weaknesses in their opponents' playing patterns and styles to maximize their earnings. Therefore, it is important to enable computer players in such games to identify flaws in opponent strategies and adapt their behaviors to exploit these flaws. This paper presents a genetic algorithm to evolve adaptive LSTM (Long Short Term Memory) poker players featuring effective opponent exploitation. Experimental results in heads-up no-limit Texas Hold'em demonstrate that adaptive LSTM players are able to obtain 40% to 1360% more earnings than cutting-edge game theoretic poker players against opponents with various flawed strategies. In addition, experimental results indicate that adaptive LSTM players evolved through playing against simple and weak rule-based opponents can achieve comparable performance against top game-theoretic poker players. The approach introduced in this paper is a promising start for building adaptive computer players for imperfect information games.

1. Introduction

In an imperfect information game, players often play against one or multiple opponents repeatedly, making it possible to identify and exploit flaws and weaknesses in the opponents' playing patterns and strategies. Poker games such as Texas Hold'em are typical examples of such imperfect information games. Players in these games usually play against a group of opponents for tens or hundreds of hands, with the goal of maximizing total earnings from all hands played. When facing strong opponents, skilled poker players make patient and deliberate moves, seeking the best opportunity for a profitable play. When a strong opponent makes a mistake, or a weak opponent with flawed strategies chips in, skilled players never hesitate to exploit the opportunity for a large gain. Based on the mistakes or flaws of their opponents, skilled poker players adjust their own strategy and make calculated moves to maximize winnings. In fact, the ability to punish mistakes and exploit weaknesses is one of the factors that contribute most to high earnings of a poker player. In this sense, the skill of a poker player, be it human or computer, should be evaluated not only by their performance against the strongest players, but also by their ability to exploit opponents with flaws and weaknesses.

In order to build computer poker players with effective opponent exploitation, two challenges must be addressed.

First, the computer players must be able to identify flaws and weaknesses of their opponents through limited observation. Typical no-limit Texas Hold'em tournaments (final table) last only for a few hundred hands. Furthermore, opponents may change their strategies, thus making the window to spot and exploit flaws and weaknesses even shorter.

Second, the computer players must be able to adapt their own strategy in real time based on observations of their opponents. Since flaws and weaknesses in various playing patterns and strategies usually require different and sometimes opposite counter-strategies for maximum exploitation, adaptation must be quick, effective, and versatile.

Classic equilibrium-based approaches do not solve either of the two challenges. Researchers have been exploring alternative approaches, including neural networks, mixed-strategy model, etc. (Lockett and Miikkulainen 2008, Ganzfried and Sandholm 2011). Applications of such opponent modeling techniques in multiple imperfect information games of different complexity yielded promising results (see Section 2 for details).

The goal of this paper is to provide an alternative paradigm for building computer agents with adaptive behaviors and effective opponent exploitation for imperfect information games. The paper introduces a genetic algorithm to evolve computer poker players controlled by Long-Short Term-Memory-based multiple-module neural networks to effectively exploit a variety of common flaws and weaknesses in poker strategy. Compared with top equilibrium-based poker players, adaptive LSTM poker players demonstrate much better performance against exploitable weak op-

ponents. In addition, they can be adjusted to achieve comparable performance against state-of-the-art computer poker players in no-limit heads-up Texas Hold'em.

The remaining sections of the paper are organized as follows. Section 2 outlines related work. Section 3 details the architecture of adaptive LSTM poker players and the genetic algorithm used to evolve them. Section 4 presents experimental results in no-limit heads-up Texas Hold'em against various exploitable players as well as a cutting edge game theoretic player, Slumbot (Jackson 2016). Section 5 points out directions for future work.

2. Related Work

There has been substantial progress in research on imperfect information games in recent years. As a typical example of such games, Texas Hold'em has been heavily studied by researchers. Computer players in many variants of the game are becoming increasingly powerful and capable of emulating high-ranking professional human players.

As a classic solution concept in game theory, Nash Equilibrium has been applied to multiple variants of Texas Hold'em with remarkable success. Counterfactual Regret Minimization (CFR) make it possible to find equilibrium-based solutions to large imperfect information games (Zinkevich et al. 2008). Heads-up limit Texas Hold'em has been weakly solved (Bowling et al. 2015). Many powerful equilibrium-based no-limit Hold'em players have been built using various techniques such as discretized betting models and potential-aware automated abstraction (Gilpin, Sandholm, and Sorensen 2008), Public Chance Sampling (Johanson et al. 2012, Jackson 2013), and distributed abstraction (Brown, Ganzfried, and Sandholm 2015).

While poker players acting according to Nash equilibriums have achieved state-of-the-art performances, they suffer from two deficiencies. First, in many complex imperfect information games, multiple Nash equilibria exist. If the opponents are not following the same equilibrium adopted by such players, performance cannot be guaranteed (Ganzfried 2016). Second, these players are unable to adapt behaviors based on opponent strategy. Therefore, they are unable to fully exploit the flaws and weaknesses of their opponents.

For two-player zero-sum games such as heads-up Texas Hold'em, there exist polynomial-time algorithms for computing an equilibrium. Also, there exists a game value that is guaranteed in expectation in the worst case by all equilibrium strategies regardless of opponent strategy (Koller, Megiddo, and von Stengel 1994). Thus, the first deficiency has relatively smaller influence in heads-up poker games. Nevertheless, the second deficiency limits performance of such equilibrium-based poker players in all variants of the game, since adapting strategies to exploit different opponent weaknesses is desirable in all formats of poker games, regardless of the number of opponents.

To address this problem, opponent exploitation mechanisms must be introduced. Researchers have proposed various methods in identifying and exploiting flaws and weaknesses in opponent strategies. Such methods include opponent clustering (Teofilo and Reis 2011), and opponent modeling by mixed strategy (Ganzfried and Sandholm 2011), Bayesian model (Finnegan et al. 2005, Ganzfried 2016), and neural networks (Lockett and Miikkulainen 2008).

In addition, researchers have been exploring non-equilibrium based approaches for Texas Hold'em such as convolutional neural networks (Yakovenko et al. 2016). Evolutionary methods have been employed to optimize topologies and weights of neural networks due to lack of labeled training data and large state space (Nicolai and Hilderman 2010).

In contrast to equilibrium-based players, players controlled by neural networks are not restricted by equilibrium strategies and can be easily integrated with various opponent models to generate adaptive strategies that exploit opponents effectively. Furthermore, complex strategies and opponent models can be compressed into network connections and weights, making it possible to represent complicated game strategies efficiently.

In the past decade, a special class of recurrent neural network, Long Short Term Memory (a.k.a. LSTM, Hocreiter and Schmidhuber 1997) networks have emerged as an effective and scalable model for a number of challenging problems related to sequential patterns (Greff et al. 2015). Since poker strategies and opponent models are essentially based on sequences of moves from different players, LSTM is directly applicable to extracting useful features and learning adaptive behaviors in poker games. Moreover, genetic algorithms can be adopted to optimize LSTM cell structures and weights (Bayer et al. 2009, Rawal and Miikkulainen 2016). Therefore, evolving adaptive LSTM poker players to maximize opponent exploitation in Texas Hold'em is a practical and promising approach.

3. Evolving Adaptive LSTM Poker Player

This section introduces an evolutionary method for building adaptive heads-up no-limit Texas Hold'em players that exploit their opponents effectively. The first subsection presents the architecture of the adaptive LSTM players. The second subsection details the method in which such players are evolved. While the approach is originally designed for heads-up no-limit Texas Hold'em, similar approaches should be applicable to building artificial agents in other imperfect information games where adaptive behaviors and opponent exploitation are desirable.

3.1. Architecture

This subsection introduces the architecture of the adaptive LSTM poker players in a bottom-up manner, starting from one of the basic functional units in the architecture, i.e. Vanilla LSTM blocks (Graves and Schmidhuber 2005). Figure 1 illustrates the structure of a Vanilla LSTM block.

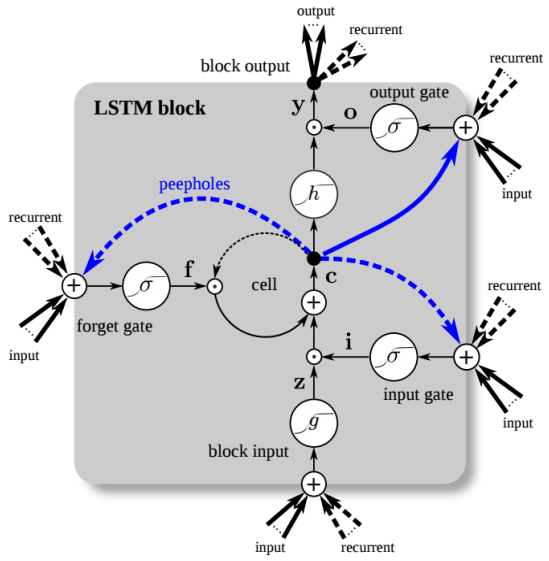

Figure 1. Vanilla LSTM Block. A block consists of three gates (input, forget, and output), a block input, a cell, an output activation function, and peephole connections. The outputs are sent back to the block input and all gates via recurrent connections.

The vector formulas for a Vanilla LSTM block forward pass (Greff and Srivastava et al, 2015) are:

$$\mathbf{z}_t = g(\mathbf{W}_z\mathbf{x}_t + \mathbf{R}_z\mathbf{y}_{t-1} + \mathbf{b}_z) \quad \text{block input (1)}$$
$$\mathbf{i}_t = \sigma(\mathbf{W}_i\mathbf{x}_t + \mathbf{R}_i\mathbf{y}_{t-1} + \mathbf{p}_i * \mathbf{c}_{t-1} + \mathbf{b}_i) \quad \text{input gate (2)}$$
$$\mathbf{f}_t = \sigma(\mathbf{W}_f\mathbf{x}_t + \mathbf{R}_f\mathbf{y}_{t-1} + \mathbf{p}_f * \mathbf{c}_{t-1} + \mathbf{b}_f) \quad \text{forget gate (3)}$$
$$\mathbf{c}_t = \mathbf{i}_t * \mathbf{z}_t + \mathbf{f}_t * \mathbf{c}_{t-1} \quad \text{cell state (4)}$$
$$\mathbf{o}_t = \sigma(\mathbf{W}_o\mathbf{x}_t + \mathbf{R}_o\mathbf{y}_{t-1} + \mathbf{p}_o * \mathbf{c}_t + \mathbf{b}_o) \quad \text{output gate (5)}$$
$$\mathbf{y}_t = \mathbf{o}_t * h(\mathbf{c}_t) \quad \text{block output (6)}$$

where $\mathbf{x}_t$ is the input vector at time t, the $\mathbf{c}$ are the cell vectors, the $\mathbf{y}$ are the output vectors. Note that the dimension of the cell vectors is equal to that of the output vectors. For convenience of discussion, it is referred to as the *cell size* in the rest of this paper. The $\mathbf{W}$ are weight matrices for the inputs, the $\mathbf{R}$ are recurrent weight matrices for the outputs, the $\mathbf{p}$ are peephole weight vectors, and the $\mathbf{b}$ are bias vectors. The asterisks denote point-wise multiplication of two vectors, and σ, g, and h are point-wise activation functions. In this application, all Vanilla LSTM blocks use hyperbolic tangent as both input and output activation function.

Vanilla LSTM blocks are organized into two modules: the game module and the opponent module. In the experiments, both modules contain a single layer of blocks. However, the structure of both modules can be adjusted for applications of different complexity.

The game module consists of ten blocks, with a cell size of five. The function of the game module is to extract useful features in the sequence of moves from both players playing their hand in a single game. Therefore, the blocks in the game module are reset to their initial state for each hand in a heads-up poker match.

The opponent module consists a single block of size ten. The opponent module is introduced to model each opponent by extracting useful features from the entire history of games played with that opponent. Hence, the blocks in the opponent module are reset for each new opponent.

Blocks in both modules receive inputs in the same format. These inputs are a compressed representation of the state of a game, including the stage of the game (i.e. preflop, flop, turn, or river), the winning probability given the board, the amount of chips committed by the opponent, the amount of chips committed by the player, and the pot odds.

The stage of the game is represented by four bits, with the bit corresponding to the current stage set to one and others zero. The winning probability is estimated assuming the opponent holding a random possible hand. The chip amounts are cumulative over each round of betting and normalized by the stack size at the beginning of the game. The pot odds o_p are computed according to

$$o_p = \frac{x}{x + s_p},$$

where x is the amount to call and s_p is the size of the pot. Pot odds are normalized into the range [0, 1). Thus, all elements of the input vector fall into the range [0, 1].

The outputs of the blocks in the game module and the opponent module are concatenated as an input vector to the decision network, which is a fully-connected feed-forward neural network with one hidden layer. The decision network uses hyperbolic tangent activation function and has a single real valued output o, which is combined with the following algorithm to decide the next move of the player.

1 Compute $y = o \cdot s_s$
2 If $y < x$, check if $x = 0$, fold otherwise
3 If $x \le y < x + r_{min}$, check if $x = 0$, call otherwise
4 Compute $k = \text{argmin}_k(|k \cdot \text{BB} - y|)$
5 Raise $k \cdot \text{BB} - x$

Algorithm 1. The Decision Algorithm. s_s is the stack size at the start of the game, x is the amount to call, BB is the big blind, r_{min} is the minimum raise, and y can be considered as the amount of extra chips the player is willing to commit.

If a call is decided but current stack is too short to call, the player will all-in instead. In the case of a raise, $k \cdot$ BB is the amount of extra chips committed to the pot, and $k \cdot$ BB - x is technically the raise amount. The actual amount of chips committed in a raise is converted to integer times of the big blind in order to merge trivially different moves. Similar to the case of a call, if current stack is too short for a decided raise, the player will go all-in.

Figure 2 presents the entire architecture of an adaptive LSTM poker player. With the support of the game module and the opponent module, the decision network is able to choose actions based on both moves in the current game and useful features extracted from previous games against the same opponent. Thus, players with the specified architecture

have the potential to adapt their behaviors for different opponents to effectively exploit the flaws and weaknesses in opponent strategy.

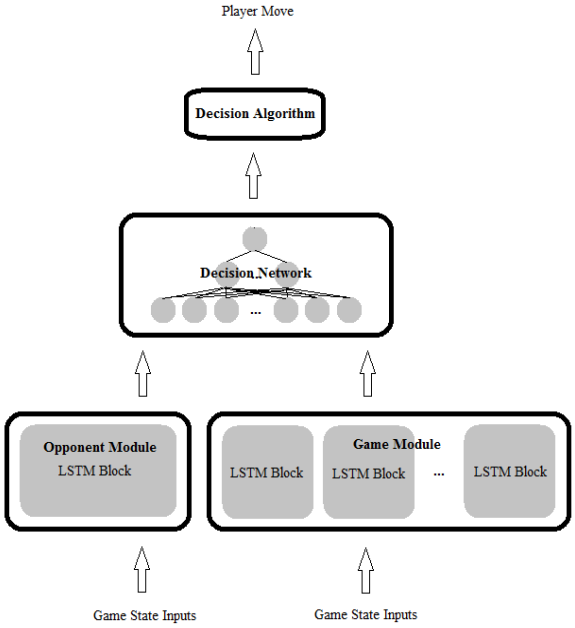

Figure 2. The Architecture of an Adaptive LSTM Player. The architecture consists of a decision algorithm and three neural network modules: the opponent module, the game module, and the decision network. The opponent module and game module contain one or multiple Vanilla LSTM blocks.

3.2. Evolution

Genetic algorithms can be specified in three parts: representation, selection, and reproduction.

Since the Vanilla LSTM blocks in the opponent module and the game module have fixed structure, and the structure of the decision network is invariant, a genetic representation of an adaptive LSTM player can be constructed by serializing and concatenating weight matrices, weight vectors, bias vectors, etc. of all components.

In order to encourage adaptation, the selection process must allow players to play against opponents with a variety of flawed strategies. These strategies should require different and preferably opposite counter-strategies for maximum exploitation. Fitness functions should evaluate a player based on its performance against all opponents. Potential bias must be taken into consideration to prevent experts specialized in exploiting certain types of opponents from constantly dominating the population. For instance, in Texas Hold'em, exploiting certain flawed strategies can easily generate far more earnings than exploiting other strategies. Hence, a fitness function that evaluates players by adding up their earnings against each opponent may undesirably favor expert players rather than adaptive players (see section 4 for more details).

Stochastic property of imperfect information games presents additional challenge to the selection process. Skilled players may suffer losses against weak opponents in short game sessions. On the other hand, prolonged game sessions may render selection painfully slow. Similarly, a small survival rate may eliminate many "unlucky" good players, but a large survival rate may preserve poor genotypes and reduce performance of future generations. Furthermore, genomes of adaptive LSTM players tend to be sizable, making it relatively difficult to discover and propagate advantageous genotypes via mutation and crossover.

To alleviate these problems, Tiered Survival and Elite Reproduction (TSER) is introduced. Survivors of a generation are divided into two tiers based on their fitness. Members of the top tier, i.e. the elites, are allowed to participate in reproduction. They are immune to mutation as long as they remain in the top tier. In contrast, members of the second tier are not allowed to reproduce, neither are they immune to mutation. However, survival provides them opportunity to preserve advantageous genotypes and improve performance through mutation.

In each generation, all players are evaluated in the same way regardless of their status. Thus, second-tier survivors from previous generations and newborn players in the latest generation can compete for positions in the top tier. After survivors are selected and classified, crossover takes place between randomly selected elites to construct children genomes. Following crossover, children and second-tier survivor genomes are mutated.

TSER provides three advantages. First, advantageous genotypes are not lost even if they perform poorly occasionally, because they can survive in the second tier. Second, the survival rate dilemma no longer exists. A high survival rate coupled with a small top tier can keep poor genotypes from being propagated while reducing accidental losses of desirable genotypes. Third, because advantageous genotypes among elites are immune to mutation, they are preserved for multiple generations, thereby providing them more time to be propagated among entire population.

To balance exploration of genome space and preservation of advantageous genotype, descending mutation rate and/or strength can be introduced. High mutation rate and strength in early stage of evolution allows more effective exploration of the genome space. As evolution proceeds, more desirable genotypes are discovered, making less aggressive mutation preferable.

While the evolutionary method outlined in this section is not limited to specific imperfect information games, experiments in Section 4 provide examples for its application.

4. Experimental Results

To evaluate the effectiveness of the method proposed in the previous section, experiments on evolving adaptive LSTM players for heads-up no-limit Hold'em are conducted. Subsection 4.1 provides details on experimental setup, including exploitable opponents, fitness evaluation, and parameter

settings. Subsection 4.2 presents and analyzes the performance of adaptive LSTM players in exploiting flawed opponents. Subsection 4.3 discusses the performance of adaptive LSTM players against Slumbot 2016, a state-of-the-art equilibrium-based player (available at *www.slumbot.com*).

4.1. Experimental Setup

As is stated in subsection 3.2, selection and fitness evaluation play a key role in evolving adaptive LSTM players to exploit opponents effectively. To encourage adaptation, players need to be evaluated by their performance against multiple opponents with different strategies. Fitness function need to reward good performance against all opponents. Parameter settings need to create a balance between genome space exploration and genotype preservation.

4.1.1. Opponents

Four rule-based players are built as opponents: the Scared Limper, the Calling Machine, the Hothead Maniac, and the Candid Statistician. Their respective strategies are outlined in algorithm 2 - 5. The variable p is the winning probability introduced in subsection 3.1.

1 If preflop on button, call the big blind (limp)
2 Compute $o = p^{50}$
3 Decide next move by *the Decision Algorithm*

Algorithm 2. The Scared Limper. "The ultra-conservative".

The Scared Limper always calls the big blind when being the small blind and folds to almost any raise at any stage of the game unless holding top hands (i.e. winning probability close to one). Counter-strategy against the Scared Limper is simple: raise any hand regardless of hand strength; if called or re-raised, fold unless holding the best hand.

1 If check is available, check, otherwise call

Algorithm 3. The Calling Machine. "Call or check only".

The Calling Machine stays in the game for a showdown regardless of hand strength. To effectively exploit its weakness, players must refrain from bluffing and raise aggressively when holding strong hands.

1 If opponent limps or checks, raise pot size
2 If raised, re-raise by twice the size of the pot

Algorithm 4. The Hothead Maniac. "Blindly raise".

The Hothead Maniac is addicted to bluffing. Similar to the Calling Machine, it is immune to bluffs. The counter-strategy must take pot size control into consideration. Folding becomes necessary when holding weak hands.

1 Compute $o = p^7$
2 Calculate move by *the Decision Algrotihm*

Algorithm 5. The Candid Statistician. "Bet honestly".

The Candid Statistician's moves always reflect its hand strength. Bluffing can be effective when the Candid Statistician holds a weak hand. In addition, the moves from the Candid Statistician can be viewed as an accurate measure of its hand strength, making it much easier for its opponent to take correct actions.

4.1.2. Evaluation and Selection

In each generation, every LSTM player plays against all four opponents. Two game sessions are played with each opponent. Both sessions contain 500 hands. Card decks in the first session are duplicated and played again in the second session. The players switch seats between the two sessions, and the memory of the LSTM player is cleared before entering the second session. At the beginning of each game, the chip stacks of the two participants are reset to \$20,000. The big blind of the game is \$100, and no ante is required. Cumulative earning (i.e. the total earning out of 1000 hands in two sessions) against every opponent is recorded for all LSTM players.

To encourage adaptation over specialization, fitness is evaluated based on Average Normalized Earnings (ANE). In the formulas below, e_{ij} is the cumulative earning of player i against opponent j, m is the number of opponents, and n_j is the normalization factor for opponent j. When the maximum cumulative earning against an opponent is less than one big blind (BB), n is set to BB to avoid normalization errors.

$$f(i) = ANE_i = \frac{1}{m}\sum_{j=1}^{m}\frac{e_{ij}}{n_j}$$

$$n_j = \max(\mathrm{BB}, \max_i(e_{ij}))$$

The maximum fitness of a player is 1.0, which means the player breaks the record of cumulative earnings against every opponent. Note that the cumulative earnings against a single opponent, regardless of its amount, contributes no more than $1/m$ to the ANE. Thus, normalization makes it more difficult for players specialized in exploiting certain opponent(s) to stay in the top survivor tier.

In the experiments presented by this section, the top tier size is not fixed. Instead, survivors of each generation are ranked according to their ANE, and the survivors whose ANE is above the average of all survivors ascend to the top tier. If the top tier contains only a single player, the second best player is added to the top tier for crossover. This mechanism allows players with genetic breakthrough to propagate advantageous genotypes quickly.

4.1.3. Parameter Settings

Table 1 presents the value of key parameters in the algorithm. Mutation rate refers to the probability for a gene (i.e. an element in the genome vector) to mutate. Mutation amount follows Gaussian distribution with zero mean, and mutation strength is the standard deviation of the distribution. Mutation rate and strength descend linearly from the initial value to the final value as evolution proceeds. Crossover is done by copying elements with odd and even index from the genome vectors of two parents, respectively.

Parameter	Value
Number of Generation	250
Population Size	50
Survival Rate	0.30
Mutation Rate (initial/final)	0.25/0.05
Mutation Strength (initial/final)	0.50/0.10

Table 1. Parameter Settings

4.2. Exploitation and Adaptation

This subsection aims at answering three questions:

- How effective are adaptive players in exploiting flaws of different opponent strategies?
- Can the evolutionary method in Section 3 evolve adaptive players with consistent performance?
- What are the specific adaptations that allow the adaptive players to exploit different opponents?

To answer these questions, the experiment introduced in the previous subsection is conducted 21 times. The champion in the last generation of each run is selected to play 500 games with each rule-based player. In addition, every rule-based player plays against one of the cutting-edge game-theoretic players, Slumbot, for 500 games. All games are in the same format as specified in the previous subsection. Table 2 compares the average performance of the champions with the performance of Slumbot. Performances are measured in mBB/hand, i.e. 1/1000 Big Blind per hand.

Depending on the opponent strategy, the average performance of the champions is 42% to 1362% better than the performance of Slumbot, making adaptive LSTM player a clear winner in the competition of opponent exploitation.

Opponent	Adaptive LSTM	Slumbot
Scared Limper	998.6 ± 2.619	702.0 ± 59.10
Calling Machine	40368 ± 1989	2761 ± 354.6
Hothead Maniac	36158 ± 1488	4988.0 ± 881.0
Candid Statistician	9800 ± 1647	4512.5 ± 471.5

Table 2. Opponent Exploitation (mBB/hand, confidence = 0.95). Adaptive LSTM players perform much better than Slumbot in exploiting different flawed opponent strategies.

Figure 3 shows boxplots of average winning per hand of the twenty-one champions against each opponent. Samples outside 1.5 IQR (Inter-quartile Range) are marked as "+". The champions from different runs demonstrate consistent performance, and the evolutionary method introduced in Section 3 is a reliable approach in evolving adaptive LSTM players with effective opponent exploitation.

The game logs of the champion with the highest fitness among all runs are analyzed to understand how adaptive LSTM players exploit each opponent. Table 3 shows some related statistics on the moves made by the champion.

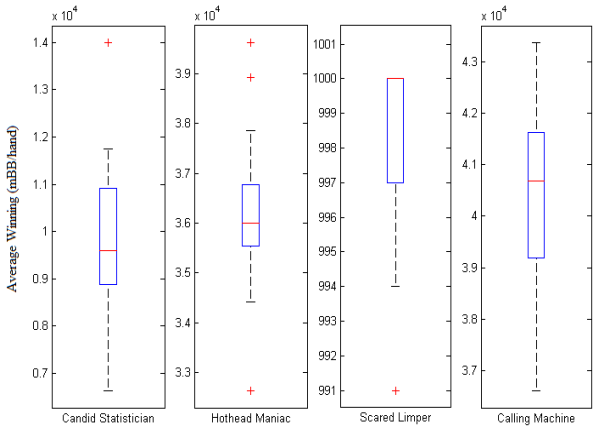

Figure 3. Champion Performance against Rule-based Players. Last-generation champions evolved in 21 runs demonstrate consistent performance against rule-based opponents.

Action frequencies are the probability of certain action being taken when they are available. Raise is available as long as a player has sufficient chips for the minimum raise. Fold is available only after opponent raises. The strategies of the Calling Machine and the Scared Limper make certain statistics unavailable (marked as "--"). The statistics in Table 3 as well as the game logs demonstrate many interesting adaptations for effective opponent exploitation.

Stage	Attribute	CS	HM	CM	SL
Preflop	Avg. Raise (BB)	2.59	3.59	3.80	2.07
	Raise Frequency	52.2%	35.5%	49.5%	99.4%
Flop	Avg. Raise (BB)	19.59	30.39	13.46	--
	Raise Frequency	30.6%	22.7%	48.4%	
Turn	Avg. Raise (BB)	63.10	63.87	66.54	--
	Raise Frequency	26.8%	11.8%	34.8%	
River	Avg. Raise (BB)	72.46	59.69	79.01	--
	Raise Frequency	10.3%	11.5%	14.0%	
Game	Fold Frequency	24.9%	4.9%	--	--
	Opponent Raise	953	1938		

Table 3. Champion Move Statistics against Different Opponents (CS = Candid Statistician, HM = Hothead Maniac, CM = Calling Machine and SL = Scared Limper, Avg. Raises are measured in Big Blinds, i.e. BB, and BB = 100)

First, the fold frequency of the champion when playing against the Candid Statistician is four times higher than the fold frequency against the Hothead Maniac. The champion adapts to the fact that the Candid Statistician's raises indicate strong hands, while the Hothead Maniac is inclined to bluff.

Second, to exploit the strategy of the Scared Limper, the champion increases preflop raise frequency to nearly 100%, forcing the extremely-conservative opponent to fold almost every hand

Third, when playing against the Hothead Maniac, the champion's raise frequency is much lower in all stages of the game. However, the average raises from the champion at the turn and the river are both close to 60 BBs. Note that a raise over 40 BBs ($4,000) generally leads to an all-in due to the Hothead Maniac's strategy and the size of the stack ($20,000). Thus, the champion exploits Hothead Maniac's strategy by folding weak hands, calling with fair hands, and inducing all-in with strong hands.

Fourth, the champion's strategy against the Calling Machine features much higher raise frequency compared to its strategy against the Hothead Maniac. Game logs indicate that the champion is not bluffing. Rather, it simply recognizes the fact that the Calling Machine's moves are independent from the strength of its hands. Therefore, if the champion's hand is better than the average, it raises. In addition, the amount of a raise is strongly correlated with hand strength, which is different from the strategy against the Candid Statistician where the champion plays strong hands less aggressively and bluffs when the opponent appears passive.

Such adaptations enable the LSTM players to make more profitable moves against different players, thus exploiting the weakness of opponent strategy effectively.

4.3. Slumbot Challenge

The adaptive players are evolved through playing against rule-based players with only simple and weak strategies. Generally, genetic algorithms tend to achieve no fitness beyond the scope of evolution. Nevertheless, adaptive players have the potential to compete against previously unseen opponents through adaptation. How well can they generalize against new opponents with much stronger strategies?

To answer this question, 500 games were played between the champions from each run (as introduced in the previous subsection) and Slumbot. The adaptive players lose to Slumbot by an average of 426 mBB per hand.

Player	Performance
Adaptive LSTM	- 426.0 ± 44.2
Adaptive LSTM, Adjusted	- 115.0 ± 11.1

Table 4. Performance against Slumbot (mBB/hand). Adaptive players evolved through playing against weak rule-based players can adapt to Slumbot and achieve comparable performance.

Game log shows that the adaptive players adopt an aggressive strategy that appears to be a mixture of the strategies against the rule-based players, and are able to bluff Slumbot effectively when holding relatively weak hands. However, Slumbot performs much better when holding strong hands. In particular, Slumbot uses value bets (i.e. small to mid-sized bets aimed at inducing a call or re-raise) effectively. In contrast, the adaptive players tend to raise rather aggressively with a strong hand, which usually reveals true hand strength and leads to a fold in response.

This problem may be caused by the fact that the adaptive players have rarely seen value bets from the rule-based opponents. Neither is value bet necessary for exploiting them. Two out of the four rule-based players never fold to any raise. Furthermore, the Candid Statistician can be quite reckless in calling or raising when the pot is small.

To alleviate this problem, two adjustments are made to the decision algorithm: (1) raise amount was restricted to be integer times of a half of the pot size, up to twice the pot size and all-in, and (2) if the desired raise amount is smaller than a fourth of the pot size, the player was made to call or check instead of raise. These rule-based adjustments limit raise amount to five options, making each option representing a much wider range of hands, which helps conceal true hand strength.

After the adjustments, the adaptive players achieve considerably better performance, losing approximately 1/10 of a big blind per hand against Slumbot. The performances in Table 4 demonstrate that adaptive LSTM players evolved via playing against weak opponents can adapt to new and strong opponents.

5. Discussion and Future Work

Experimental results in this paper demonstrate that the architecture outlined in Section 3 is effective in integrating opponent modeling with decision making. Nevertheless, the architecture is manually designed in a trial-and-error manner. The process is inefficient and unlikely to find optimal solutions. Therefore, an interesting direction for future work is to search for more effective architectures. One possible approach would be to evolve such architectures via genetic algorithms capable of optimizing multi-module neural network systems (Li and Miikkulainen 2014).

Another factor that strongly influences the performance is feature extraction. The input vector in Section 3 cannot reflect the texture of the board such as pairs and draws. On the other hand, more features result in larger models, making it more difficult to discover advantageous genotypes through evolution. Hence, a promising direction for future work is to optimize the feature set for efficient and accurate representation of the state of the game. As an example, Convolutional Neural Network (CNN) can be used to extract features from basic representation (Yakovenko et al. 2016).

In addition, experimental results in Subsection 4.3 suggest that adaptive players evolved through playing against weak opponents are able to compete against much stronger opponents. A promising direction for future work is to utilize strong opponents during evolution and evolve adaptive players that are able to identify and exploit flaws in more sophisticated opponent strategies. This way, adaptation would be based on a much stronger foundation, and adaptive players might be able to defeat Slumbot and other state-of-the-art game-theoretic players.

Since neural networks are able to compress and represent large policies, the approach introduced in this paper can be applied to imperfect information games with much larger state space such as multi-player no-limit Texas Hold'em. Further, the method may also be helpful in complicated real world applications where adaptation to partially observable environment is in need.

Conclusion

This paper introduces an LSTM-based multi-module neural network architecture that integrates opponent modeling with decision making for building adaptive players in imperfect information games. It provides a genetic algorithm, i.e. TSER, to evolve adaptive players with such architecture for effective opponent exploitation. Experimental results in heads-up no-limit Hold'em show that the proposed approach is effective and reliable in evolving computer players that exploit the weaknesses of opponent strategies through adaptation. In addition, computer players evolved by playing against simple and weak opponents can adapt its behaviors to compete against much stronger opponents and achieve comparable performances. Therefore, the approach introduced by this paper is a promising start for building adaptive computer players for large-scale imperfect information games.

Acknowledgement

This research was supported in part by NSF grants DBI-0939454 and IIS-0915038, and in part by NIH grant R01-GM105042.

References

Bayer, J., Wierstra, D., Togelius, J., and Schmidhuber, J. Evolving memory cell structures for sequence learning. *In Proc. ICANN (2)*, pages 755–764, 2009.

Bowling, M., Burch, N., Johanson, M., and Tammelin, O. Heads-up Limit Hold'em Poker is Solved. *Science*, 347 (6218), 2015.

Brown, N., Ganzfried, S., and Sandholm, T. Hierarchical Abstraction, Distributed Equilibrium Computation, and Post-Processing, with Application to a Champion No-Limit Texas Hold'em Agent. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, 2015.

Ganzfried, S. Bayesian Opponent Exploitation in Imperfect-Information Games, in *AAAI Workshop on Computer Poker and Incomplete Information*, 2016.

Ganzfried, S. and Sandholm, T. Game Theory-Based Opponent Modeling in Large Imperfect-Information Games. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems*, 2011.

Gilpin, A., and Sandholm, T. Solving Two-Person Zero-Sum Repeated Games of Incomplete Information. In *Proceedings of the Seventh International Conference on Autonomous Agents and Multi-Agent* Systems, 2008.

Gilpin, A., Sandholm, T., Sorensen, T. B. A Heads-Up No-Limit Texas Hold'em Poker Player: Discretized Betting Models and Automatically Generated Equilibrium-Finding Programs. In *Proceedings of the Seventh International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS 2008.

Graves, A., and Schmidhuber, J. Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures. *Neural Networks*, 18(5–6):602–610, 2005.

Greff, K., Srivastava, R. K., Jan, K., Steunebrink, B. R., and Schmidhuber, J. LSTM: a Search Space Odyssey. *arXiv preprint arXiv:1503.04069*, 2015.

Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. *Neural Computation 9*, 1735–1780.

Jackson, E. Solving large games with counterfactual regret minimization using sampling and distributed processing, in *AAAI Workshop on Computer Poker and Incomplete Information*, 2013.

Jackson, E. Compact CFR, in *AAAI Workshop on Computer Poker and Imperfect Information*, 2016.

Johanson, M., Bard, N., Lanctot, M., Gibson, R., and Bowling, M. Efficient Nash equilibrium approximation through Monte Carlo counterfactual regret minimization. In *Proceedings of the Seventh International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS 2012.

Koller, D., Megiddo, N., and von Stengel, B. Efficient solutions of extensive two-person games, In *Annual ACM Symposium on the Theory of Computing*, 1994.

Li, X. and Miikkulainen, R. Evolving Multimodal Behavior Through Subtask and Switch Neural Networks, *in Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems*, 2014.

Lockett, A., and Miikkulainen, R. Evolving Opponent Models for Texas Hold'em. In *Proceedings of the IEEE Conference on Computational Intelligence in Games*, IEEE Press, 2008.

Nicolai, G. and Hilderman, R. Algorithms for Evolving No-Limit Texas Hold'em Poker Playing Agents. In *Proceedings of the International Conference on Evolutionary Computation*, 2010.

Rawal, A. and Miikkulainen, R. Evolving Deep LSTMs Networks Using Information Maximization Objective, in *Proceedings of the Genetic and Evolutionary Computation Conference*, Denver, CO, 2016.

Finnegan, S., Bowling, M., Larson, B. Piccione, C., Burch, N., Billings, D., and Rayner, C. Bayes' Bluff: Opponent Modelling in Poker. In *Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence*, 2005.

Teófilo, L. F., Reis, L. P. Identifying Player's Strategies in No Limit Texas Hold'em Poker through the Analysis of Individual Moves, In *Proceedings of the 15th Portuguese Conference on Artificial Intelligence*. Lisbon, Portugal, 2011.

Yakovenko, N., Cao, L., Raffel, C., and Fan, J. Poker-CNN: A Pattern Learning Strategy for Making Draws and Bets in Poker Games, in *Proceedings of the 30th AAAI Conference on Artificial Intelligence*, 2016.

Zinkevich, M., Johanson, M., Bowling, M., and Piccione, C. Regret Minimization in Games with Incomplete Information". In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems*, NIPS, 2008.