DOI: 10.21917/ijsc.2014.0119

# A SOLUTION TO THE DOUBLE DUMMY CONTRACT BRIDGE PROBLEM INFLUENCED BY SUPERVISED LEARNING MODULE ADAPTED BY ARTIFICIAL NEURAL NETWORK

**M. Dharmalingam[1] and R. Amalraj[2]**
*Department of Computer Science, Sri Vasavi College, India*
E-mail: [1]emdharma@gmail.com and [2]r_amalraj05@yahoo.co.in

***Abstract:***

***Contract Bridge is an intellectual game which motivates multiple skills and application of prior experience and knowledge, as no player knows accurately what moves other players are capable of making. The Bridge is a game played in the presence of imperfect information, yet its strategies must be well formulated, since the outcome at any intermediate stage is solely based on the choices made during the immediately preceding phase. In this paper, we train an Artificial Neural Network architecture using sample deals and use it to estimate the number of tricks to be taken by one pair of bridge players, which is the main challenge in the Double Dummy Bridge Problem. We focus on Back Propagation Neural Network Architecture with Back Propagation Algorithm with Sigmoidal transfer functions. We used two approaches namely, High – Card Point Count System and Distribution Point Method during the bidding phase of Contract Bridge. We experimented with two sigmoidal transfer functions namely, Log Sigmoid transfer function and the Hyperbolic Tangent Sigmoid function. Results reveal that the later performs better giving lower mean squared error on the output.***

***Keywords:***

***Back Propagation Neural Network, Sigmoidal functions, Contract Bridge, Double Dummy Bridge Problem, Bidding, Playing, High – Card Point, Distribution Point Method***

## 1. INTRODUCTION

The bridge is a game which requires the application of logical, analytical and perceptive skills as well as creativity to enable intelligent decision making while making moves. Powerful approaches such as Artificial Neural Networks (ANN) are utilized to aid playing agents with well formed evaluation functions. In the game playing domain, the most popular Computational Intelligence (CI) disciplines are Artificial Neural Networks (ANN), Evolutionary Methods (EM), and Supervised Learning (SL) [1]. ANN has been effectively applied to various recognition and classification problems [2] and games [3], [4], [5]. ANNs are classified under the broad umbrella of Artificial Intelligence (AI) techniques that attempts to imitate the way a human brain works.

Among ANNs, the Back-Propagation Neural Network (BPNN) is commonly used architecture, normally trained by a supervised learning method [6], [7], [8],[9],[10]. It has been applied to solve the Contract Bridge problem using well-formulated defense models with the strongest possible assumptions about the opponent. This is used by human players because modeling the strongest opponents provides a lower bound on the pay off that can be expected when the opponents are less informed. The heuristics of beta-reduction and iterative biasing were introduced and represents the first general tree search algorithm skilled at continually performing at and above specialist level in actual card play. The effectiveness of these heuristics, when combined with payoff-reduction mini-maxing, results in iprm-beta algorithm. In solving the problems of bridge, the iprm-beta actually makes less error than human experts that produced representative solutions. [11], [33], [34].

Forward pruning techniques may produce reasonably accurate results in bridge game. Two different kinds of game trees viz., N-Game trees and N-Game like trees were used to observe, how forward pruning affects the probability of choosing the correct move. The results revealed that, mini-maxing with forward pruning did better than ordinary mini-maxing, in cases where there was a high correlation among the mini-max values of sibling nodes in a game tree. The result suggested that forward pruning may possibly be a viable decision-making technique in bridge games [12].

The Bridge Baron is generally acknowledged to be the best available commercial program for the game of Contract Bridge. The Bridge Baron program was developed by using Domain Dependent Pattern-matching Techniques which has some limitations. Hence there was a need to develop more sophisticated AI techniques to improve the performance of the Bridge Baron which was supplemented by its previously existing routines for declarer play with routine based on Hierarchical Task-Network (HTN) planning techniques. The HTN planning techniques used to develop game trees in which the number of branches at each node corresponds to the different strategies that a player might pursue rather than the different cards the player might be able to play [13].

Ginsberg's Intelligent Bridge player (GIB) is a production program, expected to play bridge at human speeds. GIB used Monte Carlo methods exclusively to select an action based on the Double Dummy analysis. All other competitive bridge-playing programs have switched their card play to similar methods, although GIB's double dummy analysis is substantially faster than most of the other programs and its play are correspondingly stronger. If the bidding simulation indicates that the opponents are about to achieve a result a great deal inferior than what they might achieve if they saw each other's cards, that is evidence that there may be a gap in the database. Unfortunately, it is also evidence that GIB is simply effectively troublesome its opponents efforts to bid accurately. GIB's bidding is substantially better than that of previous programs but not yet of expert talent [14].

The BPNN architecture with Back-Propagation Algorithm (BPA) was used in this network to train and test the data for solving Double Dummy Bridge Problems (DDGP) in Contract Bridge. In this paper, we mainly focus on two types of hand strength of human estimators during the bidding phase of Contract Bridge: (i) High – Card Point (HCP) and (ii) Distributional Point Method (DPM). This two bidding methods can be used with a

BPNN architecture to provide excellent guidance to a player without explicit use of human knowledge.

The structure of this paper is organized as follows. Section 2 gives a brief description of the Contract Bridge Game. Section 3 discusses about soft computing methods and ANNs with BPA. Our proposed model for solving the DDBP is elaborated upon in section 4. Section 5 gives implementation details. Section 6 gives experimental results and discussion. Section concludes our work and investigates future links for our research.

## 2. THE CONTRACT BRIDGE GAME

The contract bridge is a game that is played in an environment of imperfect information. There are four players in two positions of partnerships. The partners sitting opposite to each other consist of one pair [34]. It is standard to refer to the players according to their location at the table as North (N), East (E), South (S) and West (W), so N and S are partners playing next to E and W, as shown in Fig.1.

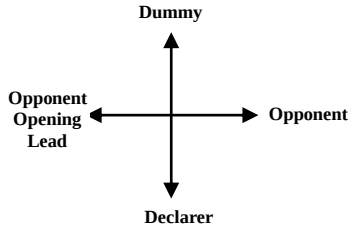


Fig.1. Game Disposition

A standard 52 cards pack is used. The cards in each suit rank from the highest to the lowest as Ace (A), King (K), Queen (Q), Jack (J), 10, 9, 8, 7, 6, 5, 4, 3, 2. The dealer deals out all the cards one at a time so that each player receives 13 of them. The game then proceeds through a bidding and playing phase. The purpose of the biding phase is to categorization of trumps and declarer of the contract. The playing stage consists of 13 tricks, with each player contributing one card to each trick in a clockwise direction with another level bid to decide who will be the declarer. The side which bids highest will try to win at least that number of tricks bid, with the specific suit as trumps. There are 5 potential trump suits: spades (♠), hearts (♥), diamonds (♦), clubs (♣) and "no-trump" which is the term for contracts played without a trump. After three following passes, the final bid becomes the contract. The team who complete the final bid will at the instant try to make the contract. The first player of this group who mentioned the significance of the contract becomes the declarer [33], [34]. The declarer's partner is well-known as the dummy shown in Fig.2.

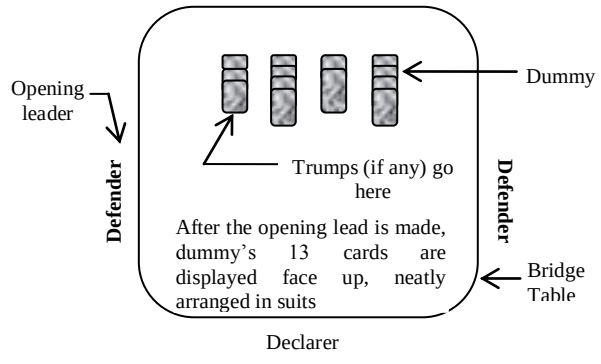


Fig.2. Bridge Table

The player to the left of the declarer leads to the first trick and instantly after this opening lead, the dummy's cards is showing. The aim of the declarer is to obtain at least the number of tricks announced during the bidding phase. The players of the opposite pair try to prevent him from doing it [15], [16]. In bridge, special focus in game representation is on the fact that players cooperate in pairs, thus distribution potentials of their hands [17].

### 2.1 THE DOUBLE DUMMY BRIDGE PROBLEM

To estimate the number of tricks to be taken by one pair of bridge players is the basis in DDBP [33]. A bridge problem is accessible for pursuit, in which the solver is presented with all four hands and is asked to determine the course of play that will achieve or defeat an exacting contract. The partners of the declarer, whose cards are placed face up on the table and played by declarer. Dummy has few rights and may not participate in choices regarding the play of the hand [34]. Estimating hands strength is a important aspect of the bidding phase of the game of bridge, since the contract bridge is a game with incomplete information and during the bidding phase. This incompleteness of information might allow for many variants of a deal in cards allocation. The player should take into account all these variants and quickly estimate the expected number of tricks to be taken in each case [18], [19].

### 2.2 THE BIDDING

The bidding phase is a conversation between two cooperating team members beside an opposing partnership. It aims to choose who will be the declarer. Each partnership uses an established bidding system to exchange information and understand the partner's bidding sequence [34]. Each player has knowledge of his own hand and any earlier bids only. A very interesting characteristic of the bidding phase is cooperation of players in a North with South and West with East. In each, player is modeled as an independent, active agent that takes part in the message process. The agent-based algorithm to use of achieve in appropriate learning, a bidding capability close to that of a human expert [20], [21], [22].

### 2.3 THE PLAYING

In the game, the play phase seems to be much less interesting than the bidding phase. ANN approaches tried to reproduce the human strategy of the play by using some tactics. The new system was able to find a strategy of play and as well a human explanation of it [23]. The player to the left of the declarer leads to the first trick and may play any card and right absent after this opening lead, the dummy's cards are exposed [34]. The play proceeds clockwise and each of the other three players in turn must, if feasible, play a card of the similar suit that the being in charge played. A player with no card of the suit led may play any card of his collection. A trick consists of four cards, one from each player, and is winning by the maximum trump in it, or if no trumps were played by the maximum card of the suit led. The winner of a trick leads to the subsequently and may lead any card. Dummy takes no lively part in the play of the hand and is not allowed to offer any advice or surveillance on the play. At any time it is dummy's turn to play, the declarer should say which of dummy's cards is to be played, and dummy plays the

card as inculcated. Finally, the scoring depends on the number of tricks taken by the declarer players and the contract [24], [25].

### 2.4 NO-TRUMP AND TRUMP-SUIT

A trick contains four cards one contributed by each player and the first player starts by most significant card, placing it face up on the table. In a clockwise direction, each player has to track suit, by playing a card of the alike suit as the one led. If a heart is lead, for instance, each player must play a heart if possible. Only if a contributor doesn't have a heart he can discard. The maximum card in the suit led wins the trick for the player who played it. This is called playing in no-trump. No-trump is the maximum ranking denomination in the bidding, in which the play earnings with no-trump suit. No-trump contracts seem to be potentially simpler than suit ones, because it is not possible to ruff a card of a high rank with a trump card. Though it simplifies the regulations, it doesn't make simpler the approach as there is no assurance that a card will take a trick, still Aces are unproductive in tricks of other suits in no-trump contracts. The success of a contract often lies in the hand making the opening lead. Hence even significant the location of all cards may occasionally be not enough to point out cards that will take tricks [17]. A card that belongs to the suit has been chosen to have the maximum value in a particular game, since a trump can be any of the cards belonging to any one of the players in the pair. The rule of the game still necessitates that if a player can track suit, the player must do so, if not a player can no longer go at the back suit, on the additional hand, a trump can be played, and the trump is superior and more commanding than any card in the suit led [18],[34].

### 2.5 HIGH CARD POINT

HCP is a system which scores 4 points for Ace, 3 points for King, 2 points for Queen and 1point for a Jack. No points are counted for 10 and below. During bidding phase of contract bridge, when a team reaches the combined score of 26 points, they should use HCP for getting final contract. There are 13 tricks in contract bridge and there is a possibility of 8 tricks by using HCP [15],[16].

### 2.6 DISTRIBUTION POINT METHOD

In the DPM system, there are score patterns which can be established in a set of cards assigned to one hand. Distribution Point Methods are awarded according to Void 3, Singleton 2 and Doubleton 1 distribution Points. An additional very important pattern which is valued is a group of honors in one suit located in the cards of both players in a pair. Having a collection of top honors in a suit allows predicting more specifically the number of tricks available in this suit [15], [16].

## 3. SOFT COMPUTING AND ARTIFICIAL NEURAL NETWORKS

With the current progress of computer technology, soft computing techniques are poised to take over from conventional deterministic computation models. The machine-intelligent performance is governed by the flexibility of the architecture, the aptitude to recognize machine incorporations of human expertise, laws of inference method and high speed of learning. All these aspects are the main constituents of the research area named Soft Computing - a more practical oriented model for solving complex real world problems [27]. Soft computing involves partnership of a number of areas, the most significant being ANNs, Fuzzy Logic (FL), Genetic Algorithm (GA) and Evolutionary Computations (EC) [6]. Among the above fields, BPNN architecture is used with BPA for solving the Double Dummy Bridge Problem in Contract Bridge.

Artificial Neural Network consists of a number of processing units which are interrelated according to some topology to reach a pattern classification task. The ANN is configured for a given application, such as pattern recognition or data classification through a learning procedure. ANNs are non-linear processing devices, which are built from prearranged elementary processing approach called neurons [28], [29]. These neurons perform basic operations on these inputs in sequence and send their response to other neurons in the network. ANN models can therefore be regarded as approximately a generalization of biological networks. In supervised learning, a supervisor is necessary for error reduced. [30], [32]. In ANNs with supervised learning, each input vector requires a matching target vector, which represents the desired output. The input vector along with the target vector is called training couple.

### 3.1 THE BACK-PROPAGATION NEURAL NETWORK ARCHITECTURE

Back-Propagation neural network architecture is a multilayer, feed-forward neural networks and it allows only for one directional indication flow. Mainly feed-forward neural networks comprise of three layers viz., an input layer, hidden layer and output layer [6]. The architecture of a BPNN is shown in Fig.3.

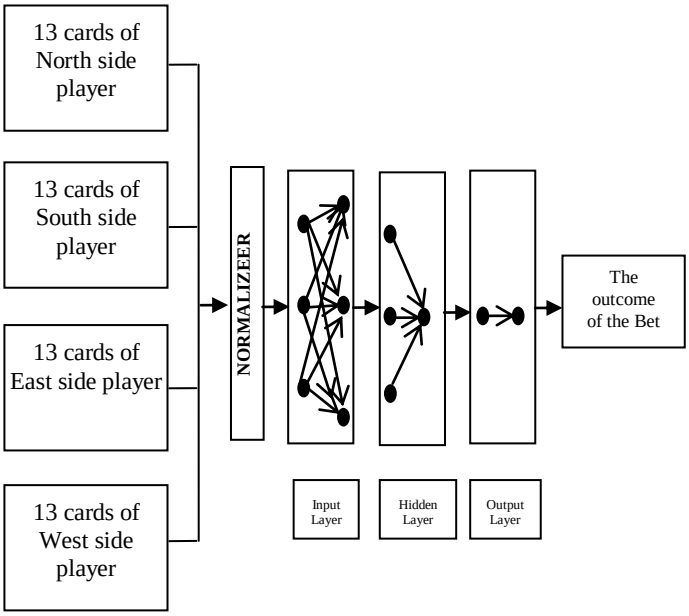


Fig.3. Architecture of BPNN

### 3.2 ACTIVATION FUNCTIONS

There are a number of common activation functions in use with neural networks. The activation function is used to decide

the output response of a neuron. The sum of the weighted input signal is applied with an activation to get the response. For neurons in same layer, same activation functions are used. There may be linear as well as non - linear activation functions [6]. The non–linear activation functions are used in a Multilayer Neural Network shown in Fig.4.

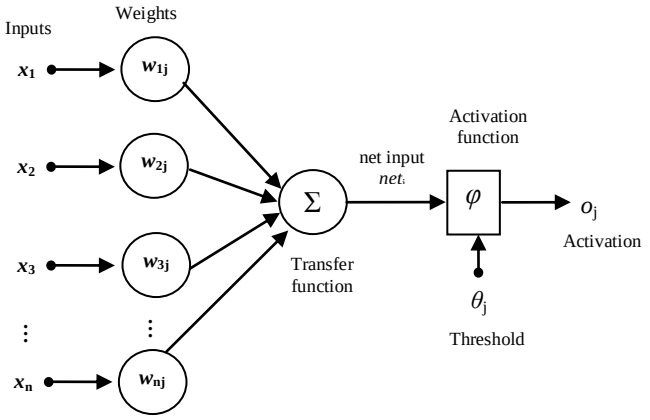


Fig.4. Activation Function

Among the above activation functions, the sigmoidal functions are widely used in Back-propagation neural networks, because of the relationship between the value of the functions at a point and the value of derivative at that point which reduce the computational weight during training. There are two types of sigmoidal functions viz., logistic sigmoid function and hyperbolic tangent function. Binary sigmoid function termed as logistic sigmoid function or unipolar sigmoid function, ranges between 0 and 1, can be defined as,

$$f(x)=\frac{1}{1+e^{-\lambda x}}$$

The bipolar sigmoid function is closely related to hyperbolic tangent function ranges between -1 and +1, defined as

$$f(x)=\frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}=\frac{1-e^{-2x}}{1+e^{-2x}}$$

## 3.3 BACK-PROPAGATION ALGORITHM

The Back-propagation neural network is a widely used type of network architecture, based on a multilayered feed forward topology, with supervised learning. This network consists of an input layer, a hidden layer, an output layer and two levels of adaptive connections. It is also fully interconnected, i.e. each neuron is connected to all the neurons in the next level. The overall idea behind back propagation is to make large change to a particular weight, '*w*' the change leads to a large reduction in the errors observed at the output nodes. Let '*y*' be a smooth function of several variables $x_i$, we want to know how to make incremental changes to initial values of each $x_i$, so as to increase the value of *y* as fast as possible. The change to each initial $x_i$ value should be in proportion to the partial derivative of *y* with respect to that particular $x_i$.

Suppose that '*y*' is a function of a several intermediate variables $x_i$ and that each $x_i$ is a function of one variable '*z*'. Also we want to know the derivative of '*y*' with respect to '*z*', using the chain rule.

$$\Delta x_i \infty \frac{\partial y}{\partial x_i} \tag{1}$$

$$\frac{dy}{dz}=\sum_i \frac{\partial y}{\partial x_i}\frac{\partial x_i}{dz}=\sum_i \frac{\partial x_i}{dz}\frac{\partial y}{\partial x_i} \tag{2}$$

The standard way of measuring performance is to pick a particular sample input and then sum up the squared error at each of the outputs. We sum over all sample inputs and add a minus sign for an overall measurement of performance that peaks at o.

$$P=-\sum_s\left(\sum_z\left[\left(d_{sz}-o_{sz}\right)^2\right]\right) \tag{3}$$

where, '*P*' is the measured performance, *S* is an index that ranges over all sample inputs, *Z* is an index that ranges overall output nodes, $d_{sz}$ is the desired output for sample input 's' at the $z^{th}$ node, $o_{sz}$ is the actual output for sample input '*s*' at the $z^{th}$ node. The performance measure *P* is a function of the weights. We can deploy the idea of gradient ascent if we can calculate the partial derivative of performance with respect to each digit. With these partial derivatives in hand, we can climb the performance hill most rapidly by altering all weights in proportion to the corresponding partial derivative. The performance is given as a sum over all sample inputs. We can compute the partial derivative of performance with respect to a particular weight by adding up the partial derivative of performance for each sample input considered separately. Each weight will be adjusted by summing the adjustments derived from each sample input [32]. We reproduce the mathematical formulation for readers' convenience. Consider the partial derivative

$$\frac{\partial P}{\partial w_{i\to j}} \tag{4}$$

where, the weight $w_{i\to j}$ is a weight connecting $i^{th}$ layer of nodes to $j^{th}$ layer of nodes. Our goal is to find an efficient way to compute the partial derivative of *P* with respect to $w_{i\to j}$ . The effect of $w_{i\to j}$ on performance, *P*, is through the intermediate variable $o_j$, the output of the $j^{th}$ node. Using the chain rule to express the derivative of *P* w.r. to

$$\frac{\partial P}{\partial w_{i\to j}}=\frac{\partial P}{\partial o_j}\frac{\partial o_j}{\partial w_{i\to j}}=\frac{\partial o_j}{\partial w_{i\to j}}\frac{\partial P}{\partial o_j} \tag{5}$$

Determine $o_j$ by adding up all the inputs to node '*j*' and passing the results through a function.

$$o_j=f\left(\sum_i 0_i w_{i\to j}\right) \tag{6}$$

where, *f* is a threshold function. Let

$$\sigma_j=\sum_i 0_i w_{i\to j}$$

We can apply the chain rule again.

$$\frac{\partial o_j}{\partial w_{i\to j}}=\frac{df\left(\sigma_j\right)}{d\sigma_j}\frac{\partial \sigma_j}{\partial w_{i\to j}} \tag{7}$$

$$\frac{\partial P}{\partial o_j}\frac{df(\sigma_j)}{d\sigma_j}o_i\frac{\partial P}{\partial o_k} \tag{8}$$

$$\frac{\partial P}{\partial w_{i\to j}} = o_i\frac{df(\sigma_j)}{d\sigma_j}\frac{\partial P}{\partial o_j} \tag{9}$$

Substituting Eq.(8) in Eq.(5), we have

$$\frac{\partial P}{\partial o_j} = \partial w_{i\to j}\frac{df(\sigma_k)}{d\sigma_k}\frac{\partial P}{\partial o_k} \tag{10}$$

$$\frac{\partial P}{\partial w_{i\to j}} = o_i\frac{df(\sigma_j)}{d\sigma_j}w_{i\to j}\frac{df(\sigma_k)}{d\sigma_k}\frac{\partial P}{\partial o_k} \tag{11}$$

Thus, the two important consequences of the above equations are, 1) The partial derivative of performance with respect to a weight depends on the partial derivative of performance with respect to the following output. 2) The partial derivative of performance with respect to one output depends on the partial derivative of performance with respect to the outputs in the next layer. The system error will be reduced if the error for each training pattern is reduced. Thus, at step '*s*+1' of the training process, the weight adjustment should be proportional to the derivative of the error measure computed on iteration '*s*' [35]. This can be written as:

$$\Delta w(s+1) = -\frac{n\partial P}{\partial w(s))} \tag{12}$$

$$\left[\Delta w(s+1) = -n\frac{\partial P}{\partial w} + a\Delta w(s)\right] \tag{13}$$

where, *n* is a constant learning coefficient and there is another possible way to improve the rate of convergence by adding some inertia or momentum to the gradient expression, accomplished by adding a fraction of the previous weight change with current weight change. The addition of such term helps to smooth out the descent path by preventing extreme changes in the gradient due to local anomalies [31].

## 4. PROPOSED ARCHITECTURE FOR SOLVING DDBP

There are several Neural Network architectures have been used to solving the DDBP. In this paper we mainly focus on Back-Propagation neural network architecture for solving the DDBP in contract bridge.

### 4.1 THE 52 (13 × 4) REPRESENTATION

In this architecture, positions of cards in the input layer were fixed, i.e. from the leftmost input neuron to the rightmost one the following cards were represented: 2♠, 3♠, . . . , K♠, A♠, 2♥, . . . , A♥, 2♦, . . . , A♦, 2♣, . . , A♣ Fig.5. This way each of the 52 input neurons was assigned to a particular card from a deck and a value presented to this neuron determined the hand to which the respective card belonged, i.e. 1.0 for North, 0.8 for South, −1.0 for West, and −0.8 for East.

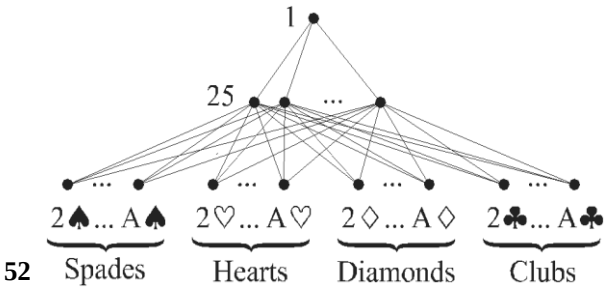


Fig.5. Neural Network Architecture with 52 input neurons

Layers were fully connected, i.e. in the 52 − 25 − 1 network all 52 input neurons where connected to all 25 hidden ones, and all hidden neurons were connected to a single output neuron.

## 5. IMPLEMENTATION

### 5.1 THE DATA REPRESENTATION OF GIB LIBRARY

The data used in this game of DDBP was taken from the Ginsberg's Intelligent Bridge (GIB) Library [14], which includes 7,00,000 deals and for each of the tricks, it provides the number of tricks to be taken by N-S pair for each combination of the trump suit and the hand which makes the opening lead. There are 20 numbers of each deal i.e. 5 trump suits by 4 sides as No-trumps, spades, Hearts, Diamonds and Clubs [26].

There are several Neural Network architectures have been used to solving the Double Dummy Bridge Problem. In this paper we focus Back-Propagation Neural Network architectures 52(13 × 4) for solving the DDBP in contract bridge. In our research for implementing GIB library data are used in MATLAB 2008a.

### 5.2 INPUT LAYER

52 cards were used in input layer. Each member was received 13 cards. The card values are determined in rank card (2, 3, K, A) and suit card (♠ (S), ♥ (H), ♦ (D), ♣(C)). The rank card is transformed using a uniform linear transformation to the range from 0.10 to 0.90. The Smallest card value is 2(0.10) and highest card value is A (0.90). The suit cards are a real number of using the following mapping: Spades (0.3), Hearts (0.5), Diamonds (0.7) and Clubs (0.9).All combination cards value rank and suit cards represented by one hand.

### 5.3 HIDDEN LAYER

There is a middle layered of hidden and internal representation 25 neuron were fully connected. The basically 4 suits, the power of trump suit, the weight of a rank card, the highest of Ace and lowest is two. The neuron representing a hand to which the card actually received input value equal 1.0. The other three neurons were assigned input values equal to 0.0.

### 5.4 OUTPUT LAYER

In this layer only one output was received and getting the result, decision boundaries were defined within the range of (0.1 to 0.9). The results were defined a priori and target of ranges

from 0 to 13 for all possible number of tricks was the use of a linear transformation. Gradient descent training function were used to train and test the data and gradient descent weight/bias learning function was used for learning the data. In this paper we compared the output data results received from these two transfer functions along with GIB library target data and got the following results in Table.1and Table.2.

Table.1. Training deals sample 20 target tricks with Log Sigmoid function and Hyperbolic Tangent Sigmoid function

| Sl. No. | GIB Target | Log Function | Hyperbolic Function |
|---|---|---|---|
| 01 | 0.75000 | 0.65477 | 0.69263 |
| 02 | 0.83000 | 0.74545 | 0.78970 |
| 03 | 1.00000 | 0.78917 | 0.89004 |
| 04 | 0.83000 | 0.82478 | 0.93428 |
| 05 | 0.75000 | 0.71576 | 0.68152 |
| 06 | 0.50000 | 0.55609 | 0.64099 |
| 07 | 0.58000 | 0.59862 | 0.61934 |
| 08 | 0.75000 | 0.90102 | 0.74347 |
| 09 | 0.50000 | 0.90383 | 0.55449 |
| 10 | 0.83000 | 0.74055 | 0.66338 |
| 11 | 0.58000 | 0.71920 | 0.59860 |
| 12 | 1.00000 | 0.88229 | 0.82769 |
| 13 | 0.58000 | 0.72812 | 0.54424 |
| 14 | 0.50000 | 0.80361 | 0.54391 |
| 15 | 0.91000 | 0.74089 | 0.85010 |
| 16 | 0.50000 | 0.96622 | 0.60367 |
| 17 | 0.50000 | 0.50643 | 0.54557 |
| 18 | 0.83000 | 0.64552 | 0.79182 |
| 19 | 0.66000 | 0.66883 | 0.59831 |
| 20 | 0.58000 | 0.60529 | 0.62150 |

Table.2. Test deals sample 10 (Even)

| Sl. No. | GIB Target | Log Function | Hyperbolic Function |
|---|---|---|---|
| 1 | 0.83000 | 0.94354 | 0.82368 |
| 2 | 0.83000 | 0.56256 | 0.71488 |
| 3 | 0.50000 | 0.70507 | 0.51046 |
| 4 | 0.75000 | 0.96712 | 0.86768 |
| 5 | 0.83000 | 0.64946 | 0.91212 |
| 6 | 1.00000 | 0.99577 | 0.99962 |
| 7 | 0.50000 | 0.56368 | 0.50053 |
| 8 | 0.50000 | 0.88346 | 0.59824 |
| 9 | 0.83000 | 0.88008 | 0.82525 |
| 10 | 0.58000 | 0.75167 | 0.57936 |

## 6. RESULTS AND DISCUSSION

In this paper sample deals data were used for training (20) and testing (10) in MATLAB 2008a. Four sides are West, North, East and South. So North and South are partners playing against East and West. The results presented in the Table.1 and Table.2 shown that the comparison of target tricks along with Log Sigmoid transfer function and Hyperbolic Tangent Sigmoid function. While comparing the trained data along with target data, the results indicated that, train and test the data shown significantly better results in both transfer functions, which minimized the total Mean Squared Error (MSE).

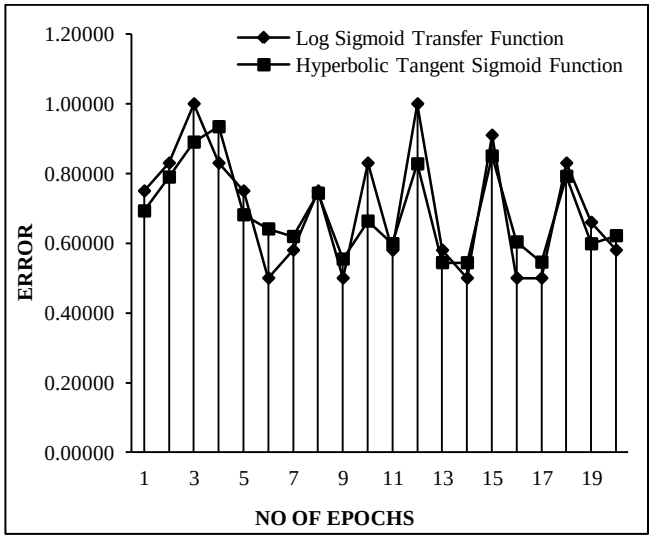


Fig.6. Training deals sampling 1000 epochs

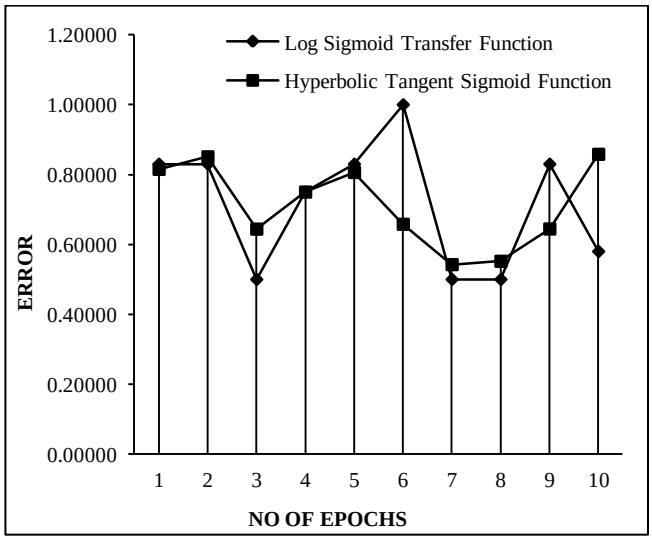


Fig.7. Testing deals sampling 1000 epochs

The Figs.6 and 7 clearly reveal that among the Log Sigmoid transfer function and Hyperbolic Tangent Sigmoid function, the later gives significantly superior results. Therefore during the bidding phase of Contract Bridge, we used a Hyperbolic Tangent Sigmoid function for generating the best HCP count system and DPM for getting the final bid.

## 7. CONCLUSION

We used ANN architectures to estimate the number of tricks to be taken by one pair of players in solving the DDBP in Contract Bridge. We employed the BPNN architecture to minimize the MSE of the output. The BPNN solved the DDBP

using two Sigmoidal functions and results showed that the Hyperbolic Tangent Sigmoid transfer function produced better results than Log Sigmoid transfer function with lower MSE. Overall, the DPM method with Hyperbolic Tangent Sigmoid function was more effective than the HCP method when taking the final bid in Contract Bridge. The superiority of the attained results strongly depends on the way a deal is coded in the input layer. In summary, the BPNN architecture proved to an excellent computational intelligence method for discovering knowledge about the game based entirely on sample training deals that appear to be specialized in solving the DDBP. The method can be used to discover new ideas in human bridge playing and help beginners and expert players in improving their bridge skills. As an extension of our current work, we are exploring new neural network architectures and hybrid algorithms to solve the DDBP more efficiently.

## REFERENCES


[1] Jacek Mandziuk, "*Knowledge-free and Learning – Based Methods in Intelligent Game Playing*", Springer, Chapter 5, pp. 53-70, 2010.

[2] Jacek Mandziuk, "Computational Intelligence in Mind Games", *Studies in Computational Intelligence*, Vol. 63, pp. 407-442, 2007.

[3] Ian Frank and David Basin, "A Theoretical and Empirical Investigation of Search in Imperfect Information Game", *Theoretical Computer Science*, Vol. 252, No. 1-2, pp. 217-256, 2001.

[4] Jacek Mandziuk, "Some thoughts on using Computational Intelligence Methods in Classical Mind Board Games", *Proceedings of the IEEE International Joint Conference on Neural Networks*, pp. 4002-4008, 2008.

[5] Jacek Mandziuk and Krzysztof Mossakowski, "Looking Inside Neural Networks Trained to Solve Double-Dummy Bridge Problems", *Proceedings of 5th Game–On International Conference on Computer Games: Artificial Intelligence, Design and Education*, pp. 182-186, 2004.

[6] S. N. Sivanandan and S. N. Deepa, "*Principles of Soft Computing*", Chapter 3, First Edition, John Wiley & Sons, pp. 59-82, 2007.

[7] Jacek Mandziuk and Krzysztof Mossakowski, "Neural Networks compete with expert human players in solving the double dummy bridge problem", *Proceedings IEEE Symposium on Computational Intelligence and Games*, pp. 117-124, 2009.

[8] Krzysztof Mossakowski and Jacek Mandziuk, "Artificial Neural Networks for solving double dummy bridge problems", *Proceedings of 7th International Conference on Artificial Intelligence and Soft Computing*, Vol. 3070, pp. 915-921, 2004.

[9] M. Sarkar, B. Yegnanarayana and D. Khemani, "Application of Neural Network in contract bridge bidding", *Proceedings of National Conference on Neural Networks and Fuzzy Systems*, pp. 144-151, 1995.

[10] M. Dharmalingam and R. Amalraj, "Neural Network Architectures for Solving the Double Dummy Bridge Problem in Contract Bridge", *Proceedings of the PSG-ACM National Conference on Intelligent Computing*, pp 31-37, 2013.

[11] Ian Frank and David Basin, "Optimal Play against Best Defence: Complexity and Heuristics", *Proceedings of the First International Conference on Computers and Games*, pp. 50-73, 1999.

[12] Stephen J. J. Smith and Dana S. Nau, "An Analysis of Forward Pruning", *Proceedings of the National Conference on Artificial Intelligence*, pp. 1386-1391, 1994.

[13] Stephen J. J. Smith, Dana S. Nau and Thomas A. Throop, "Success in Spades: Using AI Planning Techniques to Win the World Championship of Computer Bridge", *Proceedings of the National Conference on Artificial Intelligence*, pp. 1079-1086, 1998.

[14] Matthew L. Ginsberg, "GIB: Imperfect Information in a Computationally Challenging Game", *Journal of Artificial Intelligence Research*, Vol. 14, pp. 303-358, 2001.

[15] Henry Francis, A. Truscott, and D. Francis, "*The Official Encyclopedia of Bridge*", 6th Edition, American Contract Bridge League, 2001.

[16] William S. Root, "*The ABCs of Bridge*", First Edition, Three Rivers Press, 1998.

[17] Jacek Mandziuk and Krzysztof Mossakowski, "Example – based estimation of hands strength in the game of bridge with or without using explicit human knowledge", *Proceedings of IEEE Symposium on Computational Intelligence and Data Mining*, pp. 413-420, 2007.

[18] Krzysztof Mossakowski and Jacek Mandziuk, "Learning without human expertise: A case study of Double Dummy Bridge Problem", *IEEE Transactions on Neural Networks*, Vol. 20, No. 2, pp. 278-299, 2009.

[19] Stephen J. J. Smith, Dana S. Nau and T. A. Throop, "Computer Bridge: A Big Win for AI Planning", *Artificial Intelligence Magazine*, Vol. 19, No. 2, pp. 93-106, 1998.

[20] Asaf Amit and Shaul Markovitch, "Learning to bid in bridge", *Machine Learning*, Vol. 63, No. 3, pp. 287-327, 2006.

[21] T. Ando and T. Uehara, "Reasoning by agents in Computer Bridge bidding", *Proceedings of Second International Conference on Computers and Games*, pp. 346-364, 2001.

[22] T. Ando, N. Kobayashi and T. Uehara, "Cooperation and competition of agents in the auction of computer bridge", *Electronics and Communications in Japan (Part III: Fundamental Electronic Science)*, Vol. 86, No. 12, pp. 76-86, 2003.

[23] D. Khemani, "Planning with Thematic Actions", *Sadhana*, Vol. 19, No. 1, pp. 171-188, 1994.

[24] I. Frank, and D. A. Basin, "Strategies Explained", *Proceeding 5th Game programming Workshop*, pp. 1-8, 1999.

[25] Ali Awada, May Dehayni and Antoun Yaacoub, "An ATMS-Based Tool for Locating Honor Cards in Rubber Bridge", *Journal of Computing and Information Sciences*, Vol. 2, No. 5, pp. 209-218, 2011.

[26] Krzysztof Mossakowski and Jacek Mandziuk, "Neural Networks and the Estimation of Hands strength in contract bridge", *Proceedings of the 8th International Conference on Artificial Intelligence and Soft Computing*, pp. 1189-1198, 2006.

[27] B. Yegnanarayana, "*Artificial Neural Networks*", Prentice Hall of India Pvt. Ltd., Chapter 4, pp.88-141, 2010.

[28] B. Yegnanarayana, D. Khemani, and M. Sarkar, "Neural Networks for Contract Bridge Bidding", *Sadhana*, Vol. 21, No. 3, pp. 395-413, 1996.

[29] S. N. Sivanandan and M Paulraj, "*Introduction Artificial Neural Networks*", Chapter 5, pp. 119-147, 2011.

[30] M. Dharmalingam and R. Amalraj, "Supervised Learning in Imperfect Information Game", *International Journal of Advanced Research in Computer Science*, Vol. 4, No. 1, pp. 195-200, 2013.

[31] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. "Learning internal representations by error propagation", *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, Vol. 1, pp. 533-536, 1986.

[32] M. Dharmalingam and R. Amalraj, "Artificial Neural Network Architecture for Solving the Double Dummy Bridge Problem in Contract Bridge", *International Journal of Advanced Research in Computer and Communication Engineering*, Vol. 2, No. 12, pp. 4683-4691, 2013.

[33] M. Dharmalingam and R. Amalraj, "Fast Supervised learning Architecture- a work point count system coupled with Resilient Back-propagation Algorithm for Solving the Double Dummy Bridge Problem", *International Journal of Emerging Trends & Technology in Computer Science*, Vol. 3, No. 3, pp. 189-195, 2014.

[34] M. Dharmalingam and R. Amalraj, "Supervised Elman Neural Network Architecture for Solving the Double Dummy Bridge Problem in Contract Bridge", *International Journal of Science and Research*, Vol. 3, No. 6, pp. 2745-2750, 2014.

[35] J. Stuart Russell and Peter Norvig, "*Artificial Intelligence: A Modern Approach*", Second Edition, Prentice Hall, pp. 736-748, 2008.