Who should bid higher, *NS* or *WE*, in a given Bridge deal?

Jacek Mańdziuk
Faculty of Mathematics and Information Science
Warsaw University of Technology, Warsaw, Poland
mandziuk@mini.pw.edu.pl

Jakub Suchan
Faculty of Mathematics and Information Science
Warsaw University of Technology, Warsaw, Poland
kubasuchan@hotmail.com

***Abstract*—The paper proposes a neural model for a direct comparison of the two so-called Double Dummy Bridge Problem (DDBP) instances, along with a practical use-case for determining which pair, NS or WE, should propose the higher deal during a bidding phase in a Bridge game. The proposed system is composed of two identical subnetworks combined by a comparator layer placed on top of them. The base of each subnetwork is a shallow autoencoder (AE) which is further connected with a Multilayer Perceptron. The system is trained in two phases - an unsupervised one - used to create a meaningful feature-based input representation in AE compression layer, and a supervised one - meant for fine-tuning of the whole model. Training and test data are composed of pairs of Bridge deals in which the second deal in a pair is the first one rotated by 90 degrees. Since the task is to point which of the two deals promise a higher contract for the NS pair, due to deal rotation within a pair, the system effectively answers the title question "Who should bid higher, *NS* or *WE*, in a given deal?". The proposed approach is experimentally compared with two other methods: one relying on a neural system solving the DDBP and the other one employing several estimators of hand strength used by experienced players. The results clearly indicate that both neural network approaches outperform the usage of human-scoring systems by a large margin, most notably in the trump (suit) contract.**

***Index Terms*—Autoencoder, Classification, Bridge**

I. Introduction

One of the most popular and fast-growing fields on Artificial Intelligence (AI) research is concerned with various aspects of creating game AI. Such an entity would be expected to obtain promising results in a predefined competition. Initially classic board games were considered an interesting research goal including the game of checkers [19], [20], this resulted in creating one of the first self-learning programs as well as in coining the term "machine learning". Later on, another challenge in form of Chess [2], [21] was taken on with a very impressive result, beating one of the most skilled Chess players in the world. All this furnished research leading to the recent successes in the game Go [7], [22], [23]. The research domain focuses on several major problems of machine game playing, the above mentioned were all perfect-information games, but with time new challenges and ideas have arisen giving attention to partial-information games as well, in these some information is concealed from the participant, thus limiting the determinism of the outcome and the ability to fully judge the situation. As a full-fledged example of the aforementioned game type we can consider Bridge, a popular card game.

A. Rules of Bridge

Bridge is a well-known trick-taking card game which, in the variant of the so-called Contract Bridge [15] considered in this paper, requires the presence of four contestants. The participants (referred to as *North*, *East*, *South* and *West* or N, E, S, W, for short) compete in teams of two (NS vs WE) using a standard 52 card deck. Each Bridge game is composed of two phases: the *bidding* and the *play*.

1) The Bidding Phase: The first step of every game is the *bidding phase*, in which the adversaries decide about the final contract to be played. This phase requires every player - starting with the dealer and proceeding clockwise - to either pass or propose a contract. The specified contract must be higher than all previously proposed calls. Every non-passing call must include one of the five game types (♣ - Clubs, ♢ - Diamonds, ♡ - Hearts, ♠ - Spades, NT - No trump) and a number referring to the quantity of claimed tricks. A contract is considered superior if the number of trick is higher or - in case of contracts with the same specified number of tricks - when the game suit is superior. The descending order of suits is the following: NT, ♠, ♡, ♢, ♣.

2) The Play Phase: After having finalized the auction, the team which has proposed the most valuable contract is bound to attain their prior declaration. Their opponents are expected to spare no efforts in order to hinder their success. In the next phase, also known as the *play phase*, one of the team members who proposed the higher contract during bidding remains in play, the other one lays his cards down on the table face up, from this moment he no longer actively takes part in the current play phase. The opponent left to the remaining team member initiates the game by placing a card of his choice on the table. Every participant is now expected to match the suit of the first card played in the current trick, after having done so, the trick is claimed by the owner of the highest-ranked card (including the trump suit cards). The winning player is entitled to be the one to initiate the next trick.

The ultimate goal of the game is to fulfill the previously declared contract. In a single game the highest number of tricks to be claimed is limited to 13, so if x tricks were claimed by the playing pair, the opponent pair succeeds if they manage

978-1-7281-2009-6/$31.00 paper N-20098.pdf

to collect more than $13-x$ tricks. Please consult the rule book of the World Bridge Federation [15] for a detailed explanation of the game rules.

The remainder of the paper is arranged as follows. In the next section the DDBP is formally defined along with a short discussion on its practical relevance and presentation of our motivation for pursuing this research. Section III provides a brief overview of main scoring systems used by human Bridge players for an estimation of the hand's strength, which will serve as one of baseline approaches in this study. Our neural comparator of DDBP instances is presented in detail in section IV, followed by a description of the experimental setup and discussion of results in section V. Conclusions and directions for future research are addressed in the last section.

II. Double Dummy Bridge Problem

The bidding phase is without any doubts rather complex. Not only must two partners be able to estimate which cards their opponents hold but, at the same time, they also have to try to account for the strength of their partner's hand. Both team members try their best in order to indicate essential information about their hands using various calling conventions but, nevertheless, some details concerning their associate's cards remain uncovered. The above reasons contribute to the advanced difficulty and unpredictability of Bridge. On a general note, humans cope with this issue by planning, making predictions and certain assumptions about location of the most relevant cards based on the bidding phase, and finally using a great deal of experience

It is worth noting that Bridge programs contemporarily represent a similar skill level as humans although they have not been able to dominate and reach grand master levels yet [26]. With progress a few very strong Bridge programs have appeared, among others Wbridge5 [25], these programs usually rely on simulations of various scenarios which later allow to employ the Monte Carlo method and estimate the outcome of the game. This method was further refined using random seed methodology [25].

The above considerations lead directly to the Double Dummy Bridge Problem (DDBP), which consists in answering the question about how many tricks should be taken by the pair NS with all cards revealed, assuming a given trump suit (or NT game), and with perfect play of all four parties. Event though, the problem addresses the perfect-information situation (locations of all cards are known), due to its intrinsic complexity and high sensitivity to changes in card distribution (even swapping only two card between adversaries can cause the result to be vastly different), solving the DDBP poses a real challenge for Machine Learning (ML) classification methods.

As mentioned above, from a practical point of view extremely fast solving of DDBP is of paramount importance in simulation-based Bridge solvers which rely on solving its numerous instances so as to perform an accurate estimation of the expected outcome, considering various scenarios. In particular DDBP is extensively used by the *partition search algorithm* [1], [8] or *upper-confidence-bound* algorithms [10].

A. Related work

As mentioned in the previous section, the DDBP was extensively used is several Bridge playing programs, most notably the Ginsberg's Intelligent Bridge player [8] which employed partition search to support calculation of the exact DDBP solutions. Ginsberg's approach was recently revisited by Beling [1] who proposed several improvements resulting in a significant reduction of the search tree size and the calculation time.

A different approach relying on bandit-based method [11] was proposed in [10] where the problem of bidding was transformed into a learning problem and tackled accordingly using cost-sensitive classifiers and upper-confidence-bound algorithms. With respect to application of neural networks to solving DDBP, one could mention our Multilayer Perceptron (MLP) approach, which is summarized in the following section, it was followed by several recent papers of Dharmalingam and Muthusamy. In particular, in [4] the influence of the training method selection on the overall network performance is analyzed leading to a conclusion about the advantage of the Resillient Backprop (RProp) method over the classical BackProp formulation. In a subsequent paper [5] they analyze the impact of the selected activation transfer function on the network output error and experimentally demonstrated an advantage of the hyperbolic tangent function over sigmoid one. Another two papers were devoted to testing other than MLP architectures, namely the Elman network [6] and the Cascade Correlation network [18], using the RProp training algorithm in both cases. The first provides a slight insight on the advantages of the Elman network concerning time benefits as well as architecture. The latter also concerns itself with the comparison of learning algorithms confirming the conclusions presented in [4].

B. Summary of our previous approaches to DDBP

Our previous Bridge-related research was concentrated on efficiently solving the DDBP with the use of neural networks. Chronologically and methodologically it can be divided into two phases, briefly summarized below.

1) Multilayer Preceptrons: Our initial approach to solving the DDBP [16] relied on testing several Multilayer Perceptron (MLP) approaches with the research focus gradually shifting from the problem of efficient architecture selection (in terms of input deal representation and connection weight layout), through comparison of MLP performance with that of world-class human players in solving the DDBP, to the problem of explainability of the Bridge related knowledge acquired by the networks during the training process. Partial outcomes referring to the above-mentioned aspects have been published in several papers and summarized in [17].

Our initial concern was the selection of an efficient neural architecture (among the feed-forward MLP networks) in terms of the number of layers, weight connection pattern and, above all, suitable deal representation in the input layer. In extensive tests it was proven that the best encoding among several tested possibilities was the one that assigned 52 neurons to each hand

and explicitly pointed a relevant subset of 13 cards belonging to each of them. The reason for a superior performance of this particular representation should be attributed to the high sensitivity of the classification results with respect to particular location of main cards, such as honors (AKDJT) and trump cards. The resulting input layer was composed of 208 neurons out of which 52 inputs were active (equal to 1) and the remaining were equal to 0. Apparently, only in the case of explicitly pointing out the location of these cards in a highly redundant but at the same time straightforward way, the network could properly handle the deal information presented as the input (please consult [17] for a more detailed discussion). The best MLP architecture found during our experiments was composed of 4 layers $208 - 52 - 13 - 1$ or, more precisely, $52x4 - 13x4 - 13 - 1$, i.e. with dedicated connections between the input and the first hidden layer (1hl) to make independent hand representations in the 1hl. The network has full connections between the 1hl and the 2hl, and between 2hl and the output. A single sigmoid output neuron predicted the number of tricks to be taken by NS - its output range was divided into 14 sub-intervals of equal length representing 14 possible answers. This winning architecture is depicted in Figure 1. This architecture trained on 100 000

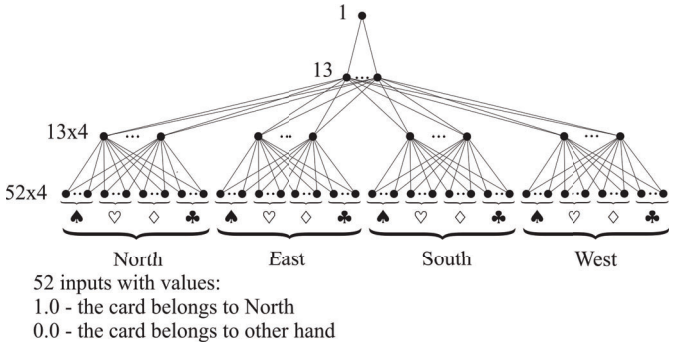

Fig. 1. Network and input encoding from a previous approach

deals and tested on another set of 100 000 deals was able to accomplish the accuracy of 53.11% for suit contracts and 37. 80% for NT (No Trump) ones [13]. In order to assess these outcomes a comparative experiment was run among world class Bridge players who took part via Internet. A group of 10 internationally recognized players (4 Grand Masters, 3 International Masters and 3 Masters), on a subset of testing deals accomplished the level of 53.06% and 73.68% for suit and NT contracts, respectively. While in the case of NT deals the human dominance in undisputable, for suit deals they are on par with out MLP approach. The reasons for this results discrepancy between suit and NT contracts are discussed in detail in [13].

During experiments we have analyzed the weight patterns in the network, we have seen the network concentrate on honor connections, a very clear case were the input neurons representing Aces ($A\spadesuit, A\heartsuit, A\diamondsuit, A\clubsuit$), where the absolute value of the connection was clearly the largest. Another interesting phenomenon were neurons specialized in honors in a specific suit, as well as those "dedicated" to one suit in general [12], [17].

2) Baseline training with autoencoders (AE-MLP approach): As over the years the computation capabilities of everyday-use computers have drastically increased more complicated models have become possible, this essentially led to the growth of popularity of deep-learning models. Taking this into account we revisited the DDBP problem by devising another neural approach, relying on the compression abilities of autoencoders (AE). The AE's capability to extract distinctive features allowed to reduce the dimensionality of the problem and in several experiments allowed to slightly surpass the previously achieved above-mentioned MLP-based approach [14], [24].

In short, the method (abbreviated as AE-MLP) employed shallow AEs during an unsupervised pretraining phase and Multilayer Perceptron networks (MLPs) with three hidden layers, built on top of these trained AEs, in the final fine-tuning training phase. The AE networks were composed of 3 layers: the input layer with 208 neurons using deal encoding presented in Figure 1, a compression layer composed of 104 or 156 units, and an output layer with 208 neurons. Once the training of AE was completed the output layer was discarded and the first two layers (input and compression one) were used as the base of the final network architecture, which underwent final, fine-tuning training. This final architecture, besides the two AE layers, was composed of 3 additional fully connected hidden layers with 52 and 13 units, resp. The output layer was composed of either 1 neuron (with the output range divided into 14 subintervals) or 14 neurons with a softmax function. In either case the output represented one of the 14 possible classes - the number of tricks to be taken by pair NS, from 0 to 13.

Two variants of AE architectures were considered in [14] which differed by the topology of connections between the input and the compression layer. In the baseline case the layers were fully connected leading to a $208 - 156/104 - 208$ architecture. In the other case, each 52-neuron representation of a given hand in the input layer was fully connected with $1/4$ of the compression layer neurons (26 or 39 depending on the particular setup), without any connections to other compression units. This way in the compression layer each 52-neuron hand representation was transformed to a 26 or 39 unit one. The network architectures of both types are presented in Figures 2 and 3, and denoted as **F** (fully connected) and **D** (dedicated connections), resp.

In the DDBP experiments reported in [14] the network of type D appeared to be slightly superior to its F counterpart, achieving the final accuracy of 51.28% and 41.73%, respectively for suit and NT contracts. In both cases the compression layer with 104 units outperformed slightly the 156 unit version.

C. Motivation and research goals

One practically-relevant aspect of Bridge research is the nature of Bridge tournaments in which 4-player teams are most often considered, and each deal is simultaneously played twice

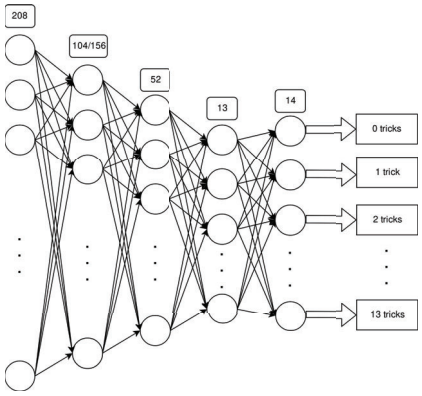

Fig. 2. AE network of type F (fully connected). The size of the 1hl is equal to 104 or 156, depending on the experimental setup

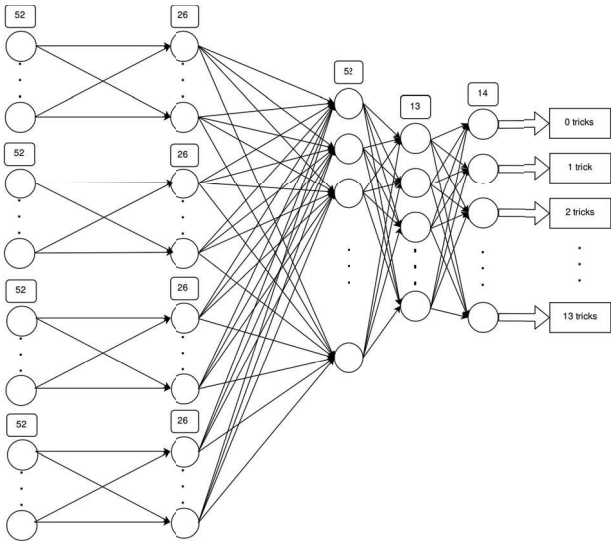

Fig. 3. AE network of type D (dedicated connections). The size of the 1hl is equal to 104 or 156, depending on the experimental setup

- on one table in the line (NS) by one pair of the team and on the other table in the line WE - by the other pair from the team. This way the team's performance becomes independent on the "luck factor" related to the strength of each hand in (randomly) selected deal. This leads to a question which line (NS or WE) is stronger and consequently which pair (NS or WE) is more likely to play it. This question can be approached in the three straightforward ways, which are experimentally verified and compared is this work.

The first one is a straightforward extension of our recent approach to solving DDBP [14], [24] which relies on using the AE networks and is briefly summarized above. Since the solution to the DDBP answers the question about the number of tricks to be taken by pair NS one can easily check if it is greater than 6, and if so - appoint NS as the stronger pair, or appoint WE, otherwise.

The second possibility is to train a classifier to point which of the two deals presented in the input promises stronger contract on NS line. Application of such a classifier to a pair composed of an original deal and the same deal rotated by 90 degrees will directly answer the question whether the stronger line is NS (if the original deal is selected by the classifier) or WE (otherwise). This latter approach (a direct comparison) is inspired by DeepChess approach [3] which was successfully applied to compare two Chess board positions and is described in more detail in section IV.

The third option is to use the human hand-strength estimators which are applied by experienced Bridge players, and sum up the hand points separately in pairs NS and WE. A pair with a higher assessment - by means of human estimators - is a stronger one. The most commonly used human hand strength estimators are discussed in the next section.

In summary, our research goals are twofold:

- a comparison between **indirectly** addressing the above mentioned problem, by means of using the specifically designed AE+MLP architecture for the estimation of the number of tricks to be taken by the pair NS, with addressing it **directly**, by means of a comparison of two deals, one of which is a rotated version of the other;
- a comparison of both above mentioned methods with the hand strength estimators commonly used by professional Bridge players.

III. Bridge trick estimation methods

Generally speaking the methods of hand strength assessment used by human Bridge players are relatively simple and mostly base on assigning points to higher-ranked cards (*Point Count methods*) and adding correctional points for long / short suits (*Distributional Count methods*).

A. Points based methods

The base methods reward hands with high cards, known as *honors*, which are in the range from A to 10. The most popular point-based systems are presented in Table I. The Work Point Count (WPC) is the absolute leader among them.

TABLE I
Point-based methods

Method	A	K	Q	J	10
Work Point Count (WPC)	4	3	2	1	0
Bamberger Point Count (BPC)	7	5	3	1	0
Collet Point Count (CPC)	4	3	2	0.5	0.5
AKQ Points (AKQ)	4	3	2	0	0

B. Distribution based methods

The WPC method is usually used as a stand-alone approach in NT contracts (additionally assuming that the suits on the playing hands are appropriately "balanced"). In the suit contracts WPC is often extended by introduction of some correctional points which take into account suits distributions. Several popular distributional methods are mentioned below.

1) Assert system (AS): This method looks for voids (suits without any cards) and singletons (one card in a given suit) in a hand, respectively contributing another +2 and +1 to the WPC score. For each longer suit (with more than 4 cards) an additional +1 is added to the result as well.

2) Rule of three and four (R34): The rule of three and four is an extension of the WPC that specifically considers suit lengths. For each card, starting from the 5th in the trump suit, an additional point is added. The same is also done for non-trump suits, but in that case the rewarding starts already from the 4th card.

3) Plus value (PV): This method looks for honors in a hand, and for every Ace an extra value of 0.25 becomes part of the final estimation, a further bonus of 0.5 is foreseen for a 10 with any other honor or with a 9. Finally, a supplementary value of 0.5 is accounted for every suit with 3 honors or 2 out of the top 3 ones (AKQ).

C. Sample deals

In order to illustrate the human hand strength estimation methods in practice let us consider two sample deals depicted in Figures 4 and 5. Point-based evaluation of both of them is presented in Table II.

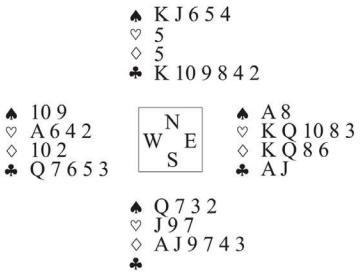

Fig. 4. First sample deal - trump suit is ♠

TABLE II
POINT-BASED AND DISTRIBUTION-BASED EVALUATION OF SAMPLE DEALS FROM FIGURES 4 AND 5. IN PARENTHESES THERE ARE BONUSES FROM THE RESPECTIVE DISTRIBUTIONAL METHODS. PLEASE NOTE THAT IN EACH METHOD EACH HAND IS *scored separately* AND ONLY THEN THE RESULTS ARE COMBINED INTO A PAIR ASSESSMENT

	Deal	Fig 4	Deal	Fig 5
System	NS	WE	NS	WE
WPC	15	25	25	15
BPC	23	41	40	24
CPC	14	26	24.5	15.5
AKQ	12	24	23	13
AS	22 (+7)	27 (+2)	30 (+5)	19 (+4)
R34	22 (+7)	31 (+6)	32 (+7)	20 (+5)
PV	15.75 (+0.75)	27.75 (+2.75)	27 (+2)	17 (+2)

In the deal from Figure 4, when taking AS into account, one has to add $+1$ for every singleton in N hand, another $+2$ for the void in S hand and a total of $+3$ for suits longer than 4 cards ($\spadesuit, \clubsuit, \diamondsuit$). For WE there are two point for long suits ($\heartsuit, \clubsuit$). When considering the R34 and NS, one has to contribute $+1$ for the 5th card in the trump suit ($\spadesuit$) and 6 points for the remaining long suits. For WE the bonus equals 6. For PV and NS one adds 0.25 for $A\diamondsuit$ and 0.5 for $(10\clubsuit, K\clubsuit)$. For WE, 0.75 for $A\heartsuit, A\spadesuit, A\clubsuit$, 1 for $(10\spadesuit, 9\spadesuit)$ and $(10\heartsuit, K\heartsuit/Q\heartsuit)$ and 1 for the two of the highest honors (twice: in $\heartsuit$ and in $\diamondsuit$).

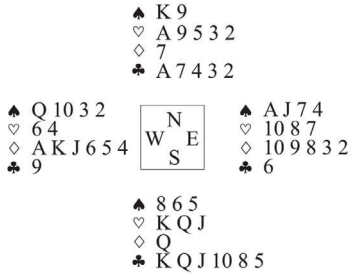

Fig. 5. Second sample deal - trump suit is ♠

Regarding the deal presented in Figure 5, applying AS will result in 2 points for a single card in $\diamondsuit$ in both hands of pair NS and 3 for long suits ($\heartsuit, \clubsuit, \clubsuit$). For WE the bonus equals 4 - 2 for singles ($\clubsuit$ in both hands) and 2 for long suits ($\diamondsuit$ twice). The R34 will add a total of 7 for NS and 5 for WE for long suits. In PV, both teams receive 0.5 for the Aces. NS additionally receives 0.5 for $10\clubsuit$ and a honor, and 1 point for $(K\heartsuit, Q\heartsuit)$ and $(K\clubsuit, Q\clubsuit)$ honors, while WE 1 and 0.5 resp. for $(Q\spadesuit, 10\spadesuit), (10\diamondsuit, 9\diamondsuit)$ and $(A\diamondsuit, K\diamondsuit)$.

IV. ADAPTATION OF THE DEEPCHESS APPROACH

In a direct comparison approach we have adapted the idea of DeepChess [3] where two chess positions were directly compared to answer the question about which one of them is more promising for a given playing side (White or Black). Similarly to our previous work [14], the model for DeepChess is trained in two stages, first one is used to extract high level features from a given position, likewise for an AE an alternative representation is built. Then in the second stage, which is now supervised, the model learns to compare chess positions and point out which one of them is more advantageous. The concept, in the context of Bridge deals, is drafted in Figure 6. In each of the subnetworks a respective deal is processed and in the final *comparator* layer a decision is output with three possible choices: the left deal is more promising for NS, both promise the same number of tricks, the right one is more promising for NS. If the deals under comparison represent a raw deal and a deal rotated by 90 degrees, then the system actually answers the question about which pair has more promising cards, NW or WE.

Each subnetwork is built in the following way: in the initial stage an AE is pretrained in accordance with the previously mentioned principles. Hidden layers of an AE are meant to extract relevant features from the data provided, this allows for the model to focus its attention on particular preselected information which in general should have the largest impact on the final output. The compression CR is set to 2 (leading

to 104 neurons in the first hidden layer) based on a bunch of preliminary experiments from previous work [14]. The resulting encoding layers (208 and 104) are extended by adding two MLP layers on top of them (52 and 13).

Two copies (clones) of such architectures are fed together into a softmax output layer composed of 3 units, which is meant to assess which of the two deals fed into the network will end with a higher outcome for the pair NS.

Keeping in mind the conclusions drawn in our previous paper [14] we have thoroughly explored two alternative topologies for our network F and D (cf. Figures 2 and 3). The final design of the system (in version F of the weights topology) is presented in Figure 7.

A. A deal representation in the input layer

The input layer was prepared based on previously conducted experiments [17]. The most efficient representation was devoting 208 neurons to serve as binary input, this allowed the deal to be fed to the network. The input consisted of 4 major parts, every one of which belongs to one of the players provided in a preset order, i.e. 52 inputs for W, 52 input for N, 52 inputs for E and finally 52 input for S. A brief representation was already presented before, in Figure 1. Within one section every neuron corresponded to one given card from all the possibilities found in the deck in the following order: $A\spadesuit, \ldots, 2\spadesuit$, $A\heartsuit, \ldots, 2\heartsuit$, $A\diamondsuit, \ldots, 2\diamondsuit$, $A\clubsuit, \ldots, 2\clubsuit$. The cards that were assigned to a specific player were marked with a 1 as input, the ones that a player did not possess remained set to 0. This led to every card being represented in the input four times - once for every player, among those two inputs exactly one was set to 1. As aftermath there were exactly 13 active inputs in every section - equal to the number of cards in a Bridge hand.

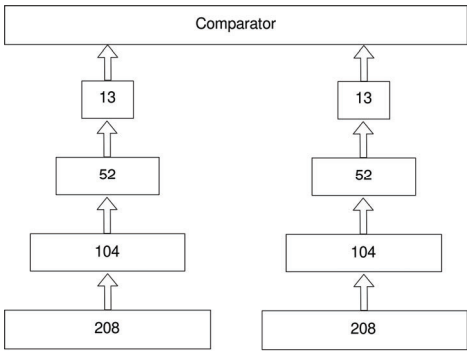

Fig. 6. Schematic presentation of a comparator-based DDBP model

V. Experimental setup and results

Similarly to our previous approaches we trained two separate models for NT and suit games, due to reference to our previous work where we have seen that the respective game strategies vary significantly.

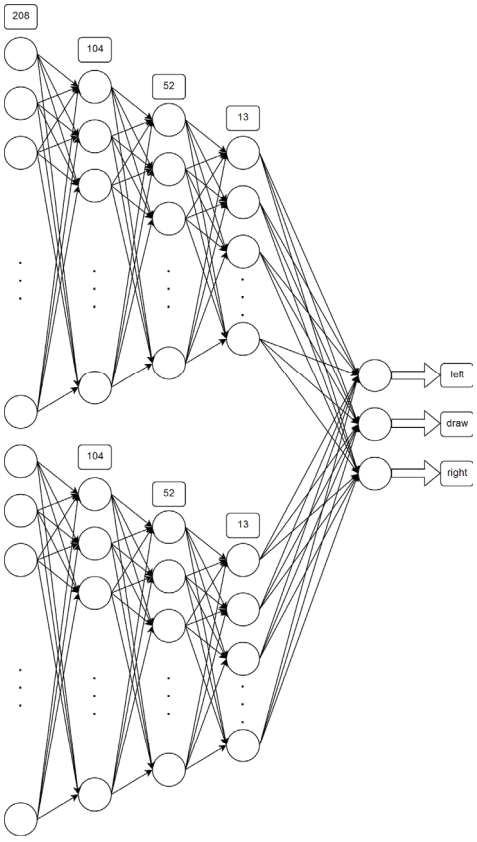

Fig. 7. Variant with 3 outputs

A. Training and testing data

All the deals used during the experiments were part of the Ginsberg's GIB Library [9] which consists of over 700 000 samples. Every sample represents a complete card distribution for every of the 4 players and is accompanied by 20 numbers representing the expected outcome for the pair NS in different scenarios. These scenarios are grouped by 5 possible game types - every suit as trump and no trump - and by the player making the opening hand, this giving a total of 20 combinations.

The size of the no trump train set was 400 000 samples thus resulting in 800 000 deals (for N and S making the initial lead), the test set consisted of 200 000 samples or a total of 400 000 deals. For trump samples the numbers were 100 000 for training and testing respectively, giving a total of 800 000 deals in each case (due to inclusion of all possible trump suits).

Training and test data were selected and converted to the appropriate format as specified in the section IV-A. In the input pairs generation process, the two paired samples needed to represent the same deal but seen as two adversaries what resulted in randomly selecting a sample and, as a paired sample, rotate it by 90 degrees (as by default the described results correspond to the pair NS). Rotating has allowed to trick the network into thinking that the other pair is now serving as the pair NS. We also had to keep in mind that the expected outcome for the game for the other pair had to be

accounted for as well, thus we obliged to subtract the expected value for NS from 13 (the total number of tricks in a game) in order to acquire the results of the game for the other.

B. Training phase 1 - training of an autoencoder

Training of the model included two major phases, the first one referred to AEs and focused on creating a simplified representation of the input. A network consisting of 208 input neurons, 104 hidden units and finally again 208 outputs was temporarily constructed for the requirements of this phase. The provisional model was trained to reproduce the input and minimize the cross-entropy error function as well as possible. The training was aborted when the change in the applied error function was below 0.01, this condition was checked every 100 iterations. Normally, the next stage would be freezing the already obtained weights, cloning the 104 unit layer and embedding the next one between the clone and its original. In our case the model was limited to just one step due to previous experience with exactly this structure [14].

Since the input and output for AE model the same cardinality the recommended error function for binary and sparse input vectors is *crossentropy* and can be described with the following formula:

$$H(x,z) = -\sum_{k=1}^{d}[x_k \log z_k + (1-x_k)\log(1-z_k)]. \quad (1)$$

where x_k is the desired output, z_k is the actual output and d being the size of a training sample. The Root Mean Square Propagation (RMSProp) algorithm with a learning rate $\eta_1 = 0.001$ was used in this phase. The value of η_1 was also taken from our previous research and experience with this methodology.

C. Training phase 2 - final training of a classifier

After having pretrained a suitable AE the model was taken into the next *fine-tuning* phase. In this phase the decoder layer was removed as it served no purpose anymore. An exact copy (clone) of the pretrained encoder part was created and both of these were finally embedded into a final model. Each encoder was connected to a suitable independent remainder - two additional layers composed of 52 and 13 units, resp. and a common output (comparator) layer. In this stage the newly created architecture was further trained to deem if one of the presented deals was more promising or in certain cases if both were equal.

D. Experimental results

Please recall from section II-C that our goal is to compare the three different ways of predicting *Who should bid higher,* NS *or* WE*, in a given deal?*. The first approach is to indirectly use our previous model from [14], [24] and its results. Please note that if comparing two pairs in the same game the total count of tricks must be equal to 13. Therefore, if the result of one team is higher or equal to 7 it means that team being in a better position. Keeping in mind that the result for the opposing team is equal to $13 - x$ one can easily compare the adversaries capabilities.

The second option is to train a DeepChess-like model introduced in section IV to point which of the two deals presented in the input promises stronger contract on NS line in the case when the second deal is a rotated (by 90 degrees) version of the first one.

The third possibility is to use the human hand-strength estimation methods presented in section III and sum up the assessment of N and S hands versus those of W and E. A higher scored pair is considered a stronger one.

Every method was tested on exactly the same set, this also was the set we have later tested our network on. The accuracy of considered approaches is presented in Table III.

TABLE III
ACCURACY OF TESTED METHODS

Method	No Trump	Trump (suit)
Work point count	80.81%	67%
Bamberger point count	82.19%	68.47%
Collet point count	82.16%	68.71%
AKQ points	79.91%	66.49%
Assert system	81.09 %	69.75%
Plus value	82.2%	69.23%
Rule of three and four	81.25%	66.58%
DeepChess-like neural system	83.42%	88.14%
AE-MLP – full connections (F)	87.18%	92.2%
AE-MLP – dedicated connections (D)	90.01	92.44%

Firstly, it can be easily observed in Table III that the results relying on human estimators are better for NT deals, what is most probably caused by the fact that the estimators mainly assign values to *individual hands*, and therefore, the synergetic effect of *combining two hands into a pair* is not taken into account. This situation is particularly harmful in the trump (suit) contracts where the exact distribution of cards on two cooperating hands is of critical importance. Neural network approaches, due to taking into account all card related information simultaneously (together) are able to infer a combined strength of a pair of players, and therefore, practically do not suffer from this problem.

Secondly, the human estimators are based on several different aspects, but they, more or less, deliver the same performance. In comparison all three neural approaches (DeepChess-like and AE-MLP ver. F and D) are clearly superior, in particular for suit contracts.

Thirdly, among neural network approaches - an indirect method, in which a deal is tested in the DDBP regime and the estimated number of tricks is compared with 6 to point out whether NS or WE has an advantageous position proved to be a few percent points stronger than a direct method relying on the DeepChess approach. This conclusions stems, most probably, from a high accuracy of AE-MLP in case one or two trick errors are allowed [14]. In the respective cases, the accuracy exceeds 95.3% and 99.7% for suit contracts and 86.1% and 96.3% for NT deals.

Finally, since both the dedicated (D) and fully-connected (F) networks acquire similar results for suit games, we belief

that there is a factor which does not depend on the topology of connections, though yet remains of cardinal importance for suit games - this could be also related to the importance of the trump suit.

Referring to two example deals presented in Figures 4 and 5, in the first one of them, in each estimation system, the pair NS is at a disadvantage, although in reality the NS pair can claim 10 tricks in ♠. On the contrary, both network approaches have chosen NS pair to be more promising. In the case of deal from Figure 5, NS will only score 3 tricks which makes the poin/distribution-based estimations inaccurate once again, although the answers of both neural network systems are correct.

VI. Summary and Conclusions

This paper proposes and experimentally evaluates three approaches to a classification problem which consists in answering the question which pair, NS or WE, has higher potential to play in a given deal. While the problem is of specific interest for the Bridge playing community, on a general note, it presents a challenging classification task, hard to solve without exhaustive simulation of the playing phase, which is avoided in this paper. The question is to be answered exclusively based on the information about a deal distribution among hands (N, E, S, W).

Generally speaking, the two neural network methods, which rely on combination of AE and MLP and apply a two-phase training regime, but differ in a way of approaching the baseline classification task visibly surpassed the third method which employs human hand strength estimators to assess the bidding potential of each pair. One of the main reasons of this phenomenon should, most probably, be attributed to the fact that unlike neural network approaches which *effectively combine* two hands into a pair, point-based systems simply sum up estimations of two hands and this way loose any synergetic effects. Such a situation is well illustrated in trump (suit) deals for which the effect of synergy is of paramount importance compared to NT deals. In the former the advantage of neural systems is enormous.

Overall, the neural networks accuracy above 90% seems to be comparable to that of the top human players. While this statement is only a hypothesis and requires experimental verification, our previous studies on a related problem (DDBP) [17] revealed that the accuracy of top human Bridge players solving this problem slightly exceeds 53.0% in the case of suit contracts and is slightly above 73.6% for the NT contracts.

Acknowledgment

This work was supported by the National Science Centre, Poland, grant number 2017/25/B/ST6/02061.

References

[1] Beling, P.: Partition search revisited. IEEE Transactions on Computational Intelligence and AI in Games 9(1), 76–87 (March 2017)

[2] Campbell, M., Hoane, A.J., Jr., F.h.H.: Deep Blue. Artificial Intelligence - Chips challenging champions: games, computers and Artificial Intelligence 134(1), 57–83 (2002)

[3] David O.E., Netanyahu N.S., W.L.: DeepChess: End-to-End Deep Neural Network for Automatic Learning in Chess. In: International Conference on Artificial Neural Networks. LNAI, vol. 9887, pp. 88–96 (2016)

[4] Dharmalingam, M., Amalraj, R.: Articifial 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, Issue 12, December 2013 2, 4683–4691 (2013)

[5] Dharmalingam, M., Amalraj, R.: A solution to the double dummy contract bridge problem influenced by supervised learning module adapted by artificial neural network. ICTACT Journal on Soft Computing 5, 836–843 (2014)

[6] Dharmalingam, M., Amalraj, R.: Supervised Elman Neural Network Architecture for Solving Double Dummy Bridge Problem in Contract Bridge. International Journal of Science and Research (IJSR) 3, 2745–2750 (2014)

[7] Gelly, S., Kocsis, L., Schoenauer, M., Sebag, M., Silver, D., Szepesvári, C., Teytaud, O.: The grand challenge of computer go: Monte carlo tree search and extensions. Commun. ACM 55(3), 106–113 (Mar 2012)

[8] Ginsberg, M.L.: http://www.gibware.com

[9] Ginsberg, M.L.: Library of double-dummy results, http://www.cirl.uoregon.edu/ginsberg/gibresearch.html

[10] Ho, C.Y., Lin, H.T.: Contract bridge bidding by learning. In: AAAI Workshop: Computer Poker and Imperfect Information (2015)

[11] Kocsis, L., Szepesvári, C.: Bandit based Monte-Carlo planning. In: Proceedings of the 17th European conference on Machine Learning. pp. 282–293. ECML'06, Springer-Verlag, Berlin, Heidelberg (2006)

[12] Mańdziuk, J., Mossakowski, K.: Looking inside neural networks trained to solve double-dummy bridge problems. In: 5th Game-On International Conference on Computer Games: Artificial Intelligence, Design and Education (CGAIDE' 2004). pp. 182–186. Reading, UK (2004)

[13] Mańdziuk, J., Mossakowski, K.: Neural networks compete with expert human players in solving the double dummy bridge problem. In: 2009 IEEE Symposium on Computational Intelligence and Games. pp. 117–124 (Sept 2009)

[14] Mańdziuk, J., Suchan, J.: Solving the Double Dummy Bridge Problem with shallow autoencoders. In: Neural Information Processing - ICONIP 2018. LNCS, vol. 11304, pp. 268–280 (2018)

[15] Manley, B., Horton, M., Greenberg-Yarbro, T., Rigal, B. (eds.): The Official Encyclopedia of Bridge. American Contract Bridge League Inc, seventh edn. (2011)

[16] Mossakowski, K., Mańdziuk, J.: Artificial neural networks for solving double dummy bridge problems. In: Artificial Intelligence and Soft Computing - ICAISC 2004. LNAI, vol. 3070, pp. 915–921 (2004)

[17] Mossakowski, K., Mańdziuk, J.: Learning Without Human Expertise: A Case Study of the Double Dummy Bridge Problem. IEEE Transactions on Neural Networks 20(2), 278–299 (2009)

[18] Muthusamy, D.: Double Dummy Bridge Problem in Contract Bridge: An Overview. Artificial Intelligence Systems and Machine Learning 10(1), 1–7 (2018)

[19] Samuel, A.L.: Some Studies in Machine Learning Using the Game of Checkers. IBM Journal of Research and Development 3(3), 210 – 229 (1959)

[20] Schaeffer, J., Björnsson, Y., Kishimoto, A., Müller, M., Lake, R., Lu, P., Sutphen, S.: Checkers is solved. Science 317, 1518–1522 (10 2007)

[21] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T.P., Simonyan, K., Hassabis, D.: Mastering chess and shogi by self-play with a general reinforcement learning algorithm. CoRR abs/1712.01815 (2017), http://arxiv.org/abs/1712.01815

[22] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., Chen, Y., Lillicrap, T., Hui, F., Sifre, L., van den Driessche, G., Graepel, T., Hassabis, D.: Mastering the game of go without human knowledge. Nature 550, 354 EP – (Oct 2017)

[23] Silver, D., Sutton, R.S., Müller, M.: Temporal-difference search in computer go. Machine Learning 87(2), 183–219 (May 2012)

[24] Suchan, J.: Deep learning in the Double Dummy Bridge Problem. Master's thesis, Warsaw University of Technology, Warsaw, Poland (2018)

[25] Ventos, V., Costel, Y., Teytaud, O., Ventos, S.T.: Boosting a bridge artificial intelligence. In: 2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI). pp. 1280–1287 (Nov 2017)

[26] Ventos, V., Teytaud, O.: Le bridge, nouveau défi de l'intelligence artificielle? Revue d'intelligence artificielle 31, 249–279 (06 2017)