ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/4376830

A Neural Network Classifier of Chess Moves

Conference Paper · October 2008

DOI: 10.1109/HIS.2008.159 · Source: IEEE Xplore

CITATIONS
3

READS
1,880

2 authors:

Cezary Dendek
Warsaw University of Technology
6 PUBLICATIONS **29** CITATIONS

SEE PROFILE

Jacek Mańdziuk
Warsaw University of Technology
214 PUBLICATIONS **2,272** CITATIONS

SEE PROFILE

All content following this page was uploaded by Cezary Dendek on 21 May 2014.

The user has requested enhancement of the downloaded file.

A Neural Network Classifier of Chess Moves

Cezary Dendek and Jacek Mańdziuk

Faculty of Mathematics and Information Science
Warsaw University of Technology
Plac Politechniki 1, 00-661 Warsaw, POLAND,
dendekc@student.mini.pw.edu.pl, mandziuk@mini.pw.edu.pl

Abstract

This paper presents an application of neural network interleaved *training algorithm proposed in [1] in the domain of chess. In order to use the referenced learning method a structure of metric space is introduced in the space of chess moves. Neural network is used as a classifier of a distance from a given move to the optimal one, leading to significant limitation of the set of moves potentially worth to be considered. The method can be used as a supportive tool in effective initial move pre-ordering which is a preliminary step in majority of search-tree methods (e.g. the efficiency of $\alpha - \beta$ pruning directly depends on the order, in which moves are considered). Proposed neural network-based classification approach can be used as a part of a hybrid AI game-tree search system.*

1. Introduction

The problem of optimal ordering of the training data has a great meaning in sequential supervised learning. It has been shown ([2],[3]), that improper order of elements in the training process can lead to catastrophic interference. This mechanism can also occur during each training epoch and disturb neural network training process. Random order of elements prevents interference but can lead to non-optimal generalization.

A new approach to patterns ordering in the context of supervised training with feed-forward neural networks has been proposed and experimentally evaluated in [1] in the context of handwritten digit recognition [4]. The idea of the method relies on *interleaving two training sequences: one of particular order and the other one chosen at random.*

Continuing the work presented in [1], it is proposed here to verify the efficacy of the interleaved training method in another application domain which is the move pre-ordering problem in the game of chess.

Certainly, in practice, efficient moves pre-ordering should rely on shallow search or no search at all, otherwise, the remaining time devoted to deeper, selective search may be insufficient. Efficacious pre-ordering of moves is a "very human" skill. Human chess players, for example, can estimate roughly 2 positions per second - compared to 200 billion ones checked in a second by Deep Blue - and therefore must be extremely effective in preliminary selection of moves.

There exist a few popular, search-free strategies of moves pre-ordering basing on historical goodness of the move in previous games played. These include the *history heuristics*, *transposition tables* or the *killer move* heuristics. In most cases these methods are highly effective since the assumption that a move which often appeared to be efficient in the past is more likely to be suitable for the current game position than other "less popular" moves is generally correct (see e.g. [5] for further discussion).

On the other hand there is still an interest in developing human-type approaches to move pre-ordering that rely on pattern-based analysis and geometrical representations. An interesting example, introduced by Zorbist [6] for Go and further developed by Greer [7] for chess, relies on heuristically defined *influence* of a particular move on certain board regions. Another geometrical method in this area was proposed by Stilman [8].

The approach introduced in this paper is also geometrically-oriented and relies on calculating a Manhattan distance between the potential target square and a particular, arbitrarily chosen predefined square ($g5$ here).

The paper is organized as follows. In the next section basic notions are defined. Sections 3 and 4 describe the set of training patterns used in the experiments and proposed training method. Numerical results of proposed *interleaved* training are presented in Sect. 5. Conclusions and directions for future research are placed in the last section.

Table 1. Fraction of samples belonging to particular classes in the test set.

dist.	number of samples [%]	dist.	number of samples [%]
0	0.48	1	2.22
2	7.38	3	12.82
4	16.82	5	17.38
6	17.23	7	14.56
8	7.64	9	2.87
10	0.51		

2. Distance definition in the space of chess moves

In order to investigate the properties of chess moves' space some basic concepts need to be introduced.

A set of chess moves E is defined as a set of all pairs $x = (x_1, x_2)$ of chessboard positions before and after performing a selected move.

A metrical space structure is imposed on the set E by introducing a *paradistance function* $P_{[\alpha,\beta]}$ defined in the following way [9]:

$$P_{[\alpha,\beta]}(x, y) = \sqrt{\alpha(B_1(x) - B_1(y))^2 + \beta(B_2(x) - B_2(y))^2}$$

where parameters α, β are real numbers satisfying the condition $\alpha, \beta > 0$ and functions $B_1(x)$ and $B_2(x)$ are respectively initial position strength and position strength change, defined by the following equation:

$$(B_1(x), B_2(x)) = (b(x_1), b(x_2) - b(x_1)) = (b(x_1), \Delta b),$$

where b is a heuristic evaluation function of chess positions.

Function P expresses the intuitive distance between chess moves: two moves are similar if initial position strengths and the moves' effects (in terms of position strength) are similar. It is then possible to compare any two moves (regardless the players' sides).

Please note, that P is not a distance function in a strict sense since

$$\sim (P(x, y) = 0 \Leftrightarrow x = y).$$

It is a consequence of defining it with the use of non-bijective function b. Definition of P implies that it is possible to divide E into subsets (abstraction classes) each of which contains only elements satisfying the condition $P = 0$.

Table 2. Distribution of classification calculated for classifiers constructed *without* **switching the training sequences.**

distance	% classifications		variance of classifier	
	1H	2H	1H	2H
0	0.59	0.58	0.0000	0.0010
1	2.05	1.96	0.0001	0.0043
2	6.55	7.61	0.0128	0.0236
3	14.06	12.72	0.0429	0.0531
4	17.62	14.82	0.0427	0.0497
5	15.69	16.04	0.0409	0.0603
6	17.50	18.16	0.0414	0.0572
7	12.55	12.94	0.0248	0.0443
8	7.52	5.80	0.0152	0.0285
9	3.12	3.10	0.0003	0.0021
10	0.44	0.41	0.0000	0.0001

3. Training patterns

Board patterns (game positions) used in the training process were selected according to the following rules:

- a move has occurred in the game played by players with ELO rating[1] at least 2300 or higher, which is roughly equivalent to the level of FIDE Master title, immediately below the International Master.
  Having such restriction imposed increases the probability that both players play suboptimal (close to optimal in the sense of move strength) strategies.

- a move has occurred between 15 and 20 game moves.
  This restriction is a consequence of an idea of classifiers localization. They are localized not only to the fragment of a chessboard (2D geometrical dimension), but also to some sphere including the time dimension. Generally, a set of used attributes might be further extended in order to decrease the complexity of decision. Independent architecture and training of classifiers plus rules of classifier choice in a given position could lead to hybrid, optimal structure covering the whole attribute set.

- a move was made by the winner of the game.
  This restriction increases a probability that a move performed in a given position was optimal, as the aim of the predictor is to model an optimal strategy.

[1] The ELO rating system is a method of calculating the relative strength of a player in popular two-player board games, e.g. chess, checkers or go - see [10] for more information.

Table 3. Quality of classification calculated for classifiers constructed *without* switching the training sequences.

distance	$P(K)$		
	1H	**2H**	**1H median**
0	0.0	0.0	0.0
1	0.0	0.0	0.0
2	0.2446	0.2717	0.4889
3	0.4000	0.5130	0.4983
4	0.2644	0.3815	0.3131
5	0.2806	0.3791	0.3011
6	0.2716	0.3122	0.2356
7	0.2379	0.2836	0.2925
8	0.1737	0.2947	0.2869
9	0.0	0.0	0.0
10	0.0	0.0	0.0
distance	$P(Y)$		
	1H	**2H**	**1H median**
0	0.0	0.0	0.0
1	0.0	0.0	0.0
2	0.1111	0.1310	0.0719
3	0.2932	0.2873	0.2876
4	0.3682	0.4061	0.3181
5	0.3273	0.5112	0.4022
6	0.4000	0.4607	0.5552
7	0.2036	0.2343	0.1540
8	0.1203	0.1170	0.1104
9	0.0	0.0	0.0
10	0.0	0.0	0.0

- a move was made by the player playing white pieces. This restriction leads to easy distinction between king and queen wing in the learning process. What is more, if the hypothesis about the existence of winning strategy for white pieces is true, it makes the classification problem (a bit) simpler task. Please note, that if prediction was to be made for a player playing black pieces then a position could be converted by switching color of the pieces and applying symmetrical reflection of the left and right sides of the board.

3.1. Positions representation

Every square of the chessboard is represented as a vector over the set $\{-1, 0, 1\}^5$. Each vector's element denotes a particular piece: rook, knight, bishop, king or pawn, whereas queen is encoded as a combination of rook and bishop. When a white piece is occupying a given square the respective vector element's value is positive; if it is a black one - the respective value is negative. If the square is unoccupied the vector associated with that square consists of all zeros. A chessboard is represented as a straightforward concatenation of all squares considered in particular, predefined order. Hence it is a vector over the set $\{-1, 0, 1\}^{5 \cdot 64}$.

Table 4. Distance of prediction result from the true one calculated for classifiers constructed *without* switching the training sequences.

radius	% of population in a sphere		
	1H	**2H**	**1H median**
0	27.31	34.51	29.21
1	53.45	65.88	55.68
2	74.16	83.18	78.52
3	86.64	93.29	89.78
4	94.55	98.71	96.48
5	98.36	99.02	99.08
6	99.74	99.93	99.93
7	99.92	99.97	100.00
8	100.00	100.00	100.00
9	100.00	100.00	100.00
10	100.00	100.00	100.00

3.2. Moves representation

The output part of the training patterns encodes a square of the chessboard on which a given move has been carried out.

Since representation of the prediction goal is usually one of the crucial problems in the design of a classifier, a lot of attention has been payed to that issue in order to maximize the chances of efficient classification.

Therefore the following factors has been taken into account:

- a dimension of an output vector should be low,
- it should be possible to combine results of classification carried out by classifiers constructed at the same or similar level of locality,
- the representation should have a structure of metrical space
- an information about possible ambiguity of prediction should be provided.

In order to satisfy the locality condition an output representation has been parameterized by a selected square of the

chessboard, denoted by S. The output space of the training patterns contains a representation of a taxicab distance from the target square of a given move to square S. A distance in the above defined set is an integer value from the interval $[0, 14]$. Consequently, an output pattern is a vector from the space $[0, 1]^{14}$, with zeros on all positions except the one denoting the encoded distance, which value is equal to 1.

3.3. Architectures of neural classifiers

Two different architectures of neural classifiers has been tested:

- multilayer perceptron with one hidden layer containing 30 neurons (denoted by **1H**),
- multilayer perceptron with two hidden layers containing 30 and 20 neurons, resp. (denoted by **2H**).

Input and output dimension of each classifier has been determined by pattern representation discussed in Sections 3.1 and 3.2.

4. Training process

The set of pattern used in the experiments consisted of 42401 elements, divided into training set T, $|T| = 38252$ and testing set V, $|V| = 4149$. All positions were sampled from chess games provided in [11].

In order to define the ordered sequence of the training patterns one of the algorithms (namely the algorithm of **Model II**) from [1] was applied. This particular model was selected in order to investigate its influence on constructed estimators. As stated in [1] other models discussed there are more effective in the case of handwritten digits recognition problem, but since the problem of optimal chess move selection is less smooth (a change of one piece position can cause not only change of move but whole player strategy), **Model II** algorithm should potentially be more successful in this case. A metric space used in calculation of the training sequence has been constructed using distance function $P_{[1,1]}$ defined in Sect. 2. Representation of output part of a pattern has been parameterized by a (randomly chosen) square board $S = g5$.

A general description of the algorithm of **Model II** is the following (see [1] for more details):

Algorithm of Model II. Given a metrics M defined on pattern space and a set $\{T_k\}$ an average pairwise distance S_n^{II} between the first n elements of the sequence can be expressed as:

$$S_n^{II} = \frac{2}{(n-1)n} \sum_{k=1}^{n} \sum_{l=k+1}^{n} M(T_k, T_l).$$

A sequence of q training patterns $(T_l)_{l=1}^{q}$ that fulfils the set of inequalities:

$$\forall_{1 \leq l \leq q-1} \qquad S_l^{II} \geq S_{l+1}^{II} \tag{1}$$

is called *ordered set of model II*.

A given set $\{T_k\}$ can be ordered to sufficiently approximate *ordered set of model II* with the use of the following algorithm:

1. Put all q elements in any sequence $(T_l)_{l=1}^{q}$.
2. Create an empty sequence O.
3. Create distance array $D[1..q]$:
$$\forall_{1 \leq l \leq q} \qquad D_l := \sum_{k=1}^{q} M(T_l, T_k)$$
4. Choose a minimal value of element of D:
$$v := min_{1 \leq l \leq q} \qquad D_l.$$
5. Pick one element k from the set $\{1 \leq l \leq q \,|\, D_l = v\}$.
6. Update distance matrix:
$$\forall_{1 \leq l \leq q} \qquad D_l := D_l - M(T_k, T_l)$$
7. Take element T_k out of sequence T and place it at the beginning of sequence O.
8. Remove element D_k from distance array.
9. Put $q := q - 1$.
10. Repeat steps 4-10 until $q = 0$.

As proposed in [1] the ordered sequence of the training patterns constructed according to the above defined algorithm was interleaved with a random sequence of these patterns. A standard backpropagation training method with learning rate and momentum equal to 0.7 and 0.1, resp. was used. Networks were trained during $1,000$ epochs. In order to check proposed algorithm two types of neural classifiers has been defined: the first one trained in standard way and the second one trained using suggested switching (interleaved) model with an initial probability of switch equal to 1, which was monotonically decreasing in time down to 0.03 in the last training epoch.

Table 5. Quality of classification calculated for classifiers constructed *with* switching of the training sequences.

distance	$P(K)$		
	1H	**2H**	**1H median**
0	0.0	0.0	0.0
1	0.0	0.0	0.0
2	0.2125	0.2843	0.2778
3	0.3778	0.5320	0.3333
4	0.2933	0.3855	0.3081
5	0.2765	0.3885	0.2422
6	0.2663	0.3199	0.2560
7	0.3066	0.3060	0.3207
8	0.1692	0.2105	0.3088
9	0.0	0.0	0.0
10	0.0	0.0	0.0
distance	$P(Y)$		
	1H	**2H**	**1H median**
0	0.0	0.0	0.0
1	0.0	0.0	0.0
2	0.0556	0.1261	0.1144
3	0.2462	0.2976	0.2989
4	0.3195	0.3809	0.3037
5	0.4133	0.5668	0.4424
6	0.4140	0.4750	0.4182
7	0.2152	0.2127	0.1258
8	0.0158	0.1029	0.1325
9	0.0	0.0	0.0
10	0.0	0.0	0.0

5. Results

5.1. Output space analysis

Descriptive statistics of output patterns belonging to the test set are presented in Table 1. It is clear from the table that classes 0 and 10 (denoting a square g5 itself and squares of distance to g5 equal to 10 are too under-represented to be learned effectively. These statistics will be used to make comparisons with statistics calculated on output set generated by a classifier based on the test set examples.

5.2. Training process without the use of ordering technique

5.2.1 Probabilities of correct classification

When assessing classifier's quality two different conditional probabilities of correct classification can be estimated. Denoting by Y the true classification of an element, by K the classification provided by a classifier, and by a the class number one can consider

- $P(K = a|Y = a)$ - a probability of correct classification to class a when element actually belongs to this class. It will be denoted as $P(Y)$.

- $P(Y = a|K = a)$ - a probability of belonging to a class selected by a classifier. It will be denoted as $P(K)$.

5.2.2 Result analysis

Some descriptive statistics calculated on the set generated by trained classifiers operating on test set are presented in Table 2. These statistics are very similar to the ones calculated for original outputs. Very low variance estimator value in case of classes 0, 1, 9 and 10 suggests that outputs representing these classes are almost constantly equal to the estimator of the expected value. The reason for such behavior is low number of patterns representing these classes.

The quality assessments of obtained classifiers are presented in Table 3. The table contains probabilities described in Sect. 5.2.1. Analysis of these results leads to hypothesis about significant differences in classification between the two tested architectures. This conclusion could be used in construction of a mixed classifier consisting of weighted classifiers of different architecture.

Due to the postulate about metrical space structure used in design of representation of the output space in Sect. 3.2 a distance from the class provided by the classifier and the true class can be calculated as a module of its difference: distance from class a to b is equal to $|a - b|$. In order to assess the quality of a classifier, a distance from a class predicted by the classifier and the true class has been calculated. Fractions of population contained in a sphere of a given radius are presented in Table 4. Basing on the presented data a relevant observation can be made: probability of classification differing from the correct one by at most 1 (i.e. selecting either the optimal square or one of its neighbors) is higher than 0.5, which is regarded as a very promising result. Moreover better results are obtained for classifiers with 2 hidden layers (denoted by **2H**). Additionally, due to unimodularity of distribution of given class' classification and the hypothesis about unbiased classifier the median combination of 3 different classifiers has been investigated. Obtained probabilities of correct classification and quality of combined classifier are higher than the ones obtained in the case of individual classifier (c.f. column **1H median** of the table).

Table 6. Distance of prediction result from true one calculated for classifiers constructed *with* switching the training sequences.

radius	% of population in a sphere		
	1H	2H	1H median
0	28.42	35.00	29.33
1	52.66	64.87	54.66
2	75.49	85.01	77.20
3	87.73	94.35	88.99
4	95.95	96.59	96.82
5	98.67	99.97	98.94
6	99.76	100.00	99.86
7	99.98	100.00	100.00
8	100.00	100.00	100.00
9	100.00	100.00	100.00
10	100.00	100.00	100.00

5.3. Training with switching the training sequence

In order to improve the quality of the classifiers a technique presented in [1] has been applied with parameters adjustment as mentioned in Sect. 4. The resulting classifiers have higher accuracy in precise prediction (cf. the values for $radius = 0$ in Tables 4 and 6), which suggests that in this case the structure of the output space is modeled more accurately than in the case of using training patterns selected in random order. Augmentation of prediction is observed in both types of classifier architectures which extends the work done in referred paper to broader class of neural networks. Worse results obtained for combined classifier can be explained by mutual dependencies between individual classifiers, which are higher in this case compared to classifiers trained with the use of randomly ordered patterns. The observed phenomenon is very promising in terms of combining both types of classifiers. Conducted research has confirmed an influence of training pattern ordering on a quality of constructed predictors.

6. Conclusions

In the paper a new approach to the problem of preselection of chess moves for further evaluation has been proposed. In the experimental evaluation of the method, classifiers with promising statistical features have been generated, supporting the conclusion about the efficacy of classifiers' localization. The results are in line with the observations presented previously in [1] in another application domain. In the future we plan to investigate the possibilities of the development of combined classifiers.

7. Acknowledgment

This work was supported by the Warsaw University of Technology under grant no. 504G 1120 0008 000. Computations were performed using grid provided by Enabling Grids for E-sciencE (EGEE) project.

References

[1] Dendek, C., Mańdziuk, J.: Including Metric Space Topology in Neural Networks Training by Ordering Patterns Proc. ICANN'06, LNCS, Springer-Verlag **4132** (2006) 644–653

[2] Ratcliff, R.: Connectionist Models of Recognition Memory: Constraints Imposed by Learning and Forgetting Functions. Psychological Review **97**(2) (1990) 285–308

[3] French, R. M.: Catastrophic Forgetting in Connectionist Networks. Trends in Cognitive Sciences **3**(4) (1999) 128–135

[4] Mańdziuk, J., Shastri, L.: Incremental Class Learning approach and its application to Handwritten Digit Recognition. Information Sciences **141**(3-4) (2002) 193–217

[5] J. Schaeffer. The history heuristic and alpha-beta search enhancements in practice. *IEEE PAMI*, 11(11):1203–1212, 1989.

[6] A. Zorbist. Feature extractions and representation for pattern recognition and the game of go. PhD Thesis, University of Wisconsin, 1970.

[7] K. Greer. Computer chess move-ordering schemes using move influence. *Artificial Intelligence*, 120:235–250, 2000.

[8] B. Stilman. *Liguistic Geometry. From search to construction*. Kluwer Academic Publishers, Boston, Dordrecht, London, 2000.

[9] Hyatt, R. M. Crafty - Computer Chess Program.

[10] Wikipedia - The Free Encyclopedia - http://en.wikipedia.prg/wiki/Elo_rating_system (2008)

[11] ChessBase - Chess Online Database - http://www.chesslive.de/ (2005)

View publication stats