Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 192 (2021) 1760–1769

Procedia Computer Science

www.elsevier.com/locate/procedia

25th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems

Chess as Sequential Data in a Chess Match Outcome Prediction Using Deep Learning with Various Chessboard Representations

Rafał Dreżewski[a,*], Grzegorz Wątor[a]

[a]*AGH University of Science and Technology, Institute of Computer Science, Cracow, Poland*

Abstract

Moves made by chess players during the match are certainly some kind of sequence. In this work, we tried to answer the question if such data could be interpreted as sequential. To achieve that, a model build with LSTM layers only was designed. Results performed by the model are justifying that thesis. A novel model architecture was also proposed and trained on multiple data types—chess moves and chess game metadata. Its purpose was to perform as high classification accuracy as possible—we managed to achieve it with a result close to 69%. Moreover, we compared a couple of chessboard representation methods, more precisely a bitmap input and algebraic input, to check which one is more relevant for the neural networks training process. Contrary to what one might suppose, better scores were reached for bitmap input, which from the theoretical point of view carries out less information than algebraic input.

Keywords: deep learning; chess; prediction; classification; LSTM; chessboard representation

1. Introduction

People can define chess in completely different ways. Some will say that it is simply a game or a sport—such opinions are mostly expressed by people who have never played chess. A chess player will say that chess is a strategy, competition, fight, great victories, but also disgraceful defeats. It cannot be denied that the two explanations of chess are not mutually exclusive. The general opinion about chess is very positive. People usually emphasize the benefits of playing chess—the improvement of memory, being more focused, and improvement in the perception effectiveness. The impact of playing chess on pupils' cognitive and academic skills—especially on their mathematical abilities—were also investigated, however the results of those experiments are not fully convincing because the placebo effect cannot be fully rulled out [10].

* Corresponding author.
E-mail address: drezew@agh.edu.pl

1877-0509
10.1016/j.procs.2021.08.180

When analyzing chess from the technical side, there is no doubt that chess is a well-structured game in which the set of all possible combinations is predefined and, above all, finite. Approximately the number of possible positions on the board is 10^{50}—it is obvious that we cannot find a solution to the chess game problem in an explicit manner using the current computer resources. Therefore, current chess engines, instead of explicitly looking for a solution, use various methods to estimate the strength of the position that takes place on the board. In the research described in this article, the problem to be solved is the assessment of this position, which boils down to the problem of predicting the result of a chess game. A chess game may end in one of three outcomes: white victory, black victory or a draw.

People who are not interested in this game might think that the latter option is the least likely. This can be the case when two low-skill players sit down to a game—they usually make so many mistakes that, as a consequence, the result is rarely a draw. For top-level games, a draw is as likely to be the result as the other two possibilities. A good example is the 2018 World Cup between Magnus Carlsen and Fabiano Caruana [1]. Of the 12 games played in standard mode, all ended in a draw.

In this paper, we will try to answer following research questions:

1. Can the moves made in a chess game be treated as sequential data?
2. How does the various kinds of chessboard conversions into vector affect the classification result?
3. What accuracy in predicting the game result will the proposed models achieve?

A series of experiments to decide whether the sequence of moves played in a chess game can be interpreted as sequential data was performed. To verify that research question, we proposed to assess how well recursive neural networks (more precisely LSTM—Long Short-Term Memory [6]) are capable of solving the problem of predicting the result of a chess game. The influence of various representations of chess positions (relevant to train the recursive neural networks) on the accuracy of classification was also investigated. The assessed representations include *bitmap input* and *algebraic input*.

Additionally, a neural network architecture was proposed that takes into account, apart from the moves played in the game, also certain metadata related to the chess game—more precisely, data of the players participating in the match, such as the names of the chess players and their current ELO chess ranking. Summing up, the network model consists of two components:

1. A neural network that classifies the outcome of a game based on data closely related to the participating chess players.
2. A neural network that classifies the result of a game according to its progress.

As a result of combining these two parts, we obtained a complete model. The exact architecture that we used, will be described in details in the next sections, but it is worth noting here that the use of LSTM (Long Short-Term Memory) layers to train a model responsible for classifying outcome of the chess game based on played moves is an innovation. Since chess seems to be a sequential game, it could be a good idea to use LSTM to build the model—these networks are great for operating on sequential data.

The proposed problem of classifying the result of a chess game, which boils down to choosing one of three options—a game won by white, a game won by black or a draw—has already been raised several times by scientists from all over the world. Some have tried to solve this problem with machine learning, others with statistical methods, and others have tried to use methods related to data mining. These solutions will be discussed below.

Scientists from the University located in the United Arab Emirates have proposed a solution that predicts the outcome of the game based on the moves made by players [8]. Their solution is based on extracting five features from each move—*Piece, Column, Row, Check*, and *Capture*. Below is a brief explanation of what each of these features means:

- Piece—the type of piece. This property has six possible values P, R, N, Q, K, representing pawn, rook, knight, bishop, queen, and king respectively.
- Column—chessboard rank, possible values are a–h.
- Row—chessboard file, possible values are 1–8.
- Capture—whether the opponent's piece has been captured or not, possible values are 0, 1.

- Check—whether the opponent's king has been checked or not, possible values are 0, 1.

The information is presented as a vector, for example: $\{N, f, 3, 0, 0, N, f, 6, 0, 0, P, c, 4, 0, 0, P, g, 6, 0, 0, \ldots\}$. Such prepared data were used to build classifiers—Naive Bayes, J48 decision tree, and SVM (Support Vector Machine). These classifiers were evaluated separately (the results are denoted as BC) and as a set of classifiers operating as a whole. In this case, two methods of combining classifiers were used: the prediction of each of them included in the composite classifier has the same weight (the authors denote this type of classifier as EE) or each of them is assigned a weight calculated according to a certain algorithm (denoted as PE).

For every classifier and competitor (BC, EE, or PE) the authors of [8] performed a number of experiments, each one with a different size of the training set (ranging from 33000 to 270000 games). All of the examined approaches showed similar trend—the larger dataset, the greater accuracy of the classification. As an example, for 33000 training data and for Naive Bayes classifier, the accuracies of PE, EE, and BC were respectively 60.2, 43.4, and 47.5. For 270000 training data, the accuracies were 62.8, 46.8, and 48.0. PE turned out to be the best ensemble learning technique with scores at least 5% higher than the ones achieved by BC, and 15% higher than those of EE. The maximum prediction accuracy that the authors of [8] managed to achieve was 65.6% for the J48 classifier and PE technique.

Zheyuan Fan, Yuming Kuang, Xiaolin Lin from Stanford University describe in [3] their method of solving the problem of chess game result prediction. The authors proposed their player ranking system, which could be better than the widely used ELO system. Based on the designed system, they predicted the results of games played by the same players. As an indicator of whether this approach is good, they used comparison between the predicted result and the actual result of the match. The dataset they used is the game results and players' Elo coefficients, taken from the internal FIDE (World Chess Federation) database and the Chessbase database. It contains data from 2000 to 2011, of which 127 months were designated as the training set, and the rest as the test set.

The authors proposed a game outcome model as the Hidden Markov Process. Further on, they presented mathematical considerations justifying the proposed solution and undertaken assumptions. Given the current strength ratings of X_i and X_j of player i and j, the prediction of the outcome of $Y_{i,j}$ is made by computing the following probabilities [3]:

$$P(Y_{i,j} = 1(white\ win)) \propto \phi_{i,j} \tag{1}$$

$$P(Y_{i,j} = 0,5(draw)) \propto C\sqrt{\phi_{i,j}(1-\phi_{i,j})} \tag{2}$$

$$P(Y_{i,j} = 0(white\ lose)) \propto 1-\phi_{i,j} \tag{3}$$

$$P(Y_{i,j} = 1(white\ win)) \propto \phi_{i,j} \tag{4}$$

The coefficients $\phi_{i,j}$ are calculated based on X_i and X_j (details of the derivation of these formulas are described in detail in [3]). The accuracy achieved by such a model is 55.63%, which is better than randomly guessing the outcome (33% chance because we have 3 possible outcomes). In the case when only the matches won by white or black (without a draw) were examined, the accuracy of predictions was 85.73%.

Diogo R. Ferreira in his work *Predicting the Outcome of Chess Games based on Historical Data* [4] proposed a similar solution to the one discussed above. A predictive model based on the Bradley-Terry probabilistic model, trained on the historical records of chess games played over the last few years, was used. The model was iterated and it included several tuning parameters. However, it is not clear what the accuracy of the proposed method's prediction was. In another interesting work, Ferreira [5] proposed a technique for determining a chess player's strength based on his actual moves in the game.

2. Methods and Algorithms

In this section, the main methods, algorithms, and tools used in the research conducted in this work will be presented. The designed models of neural networks will be discussed, then the dataset on which the models were trained and evaluated will be presented. A crucial issue in this section will also be the description of two different ways of mapping chess positions to vector form.

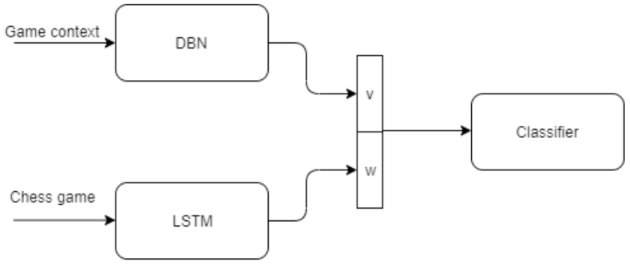


Fig. 1. General diagram of model Context Match Fusion.

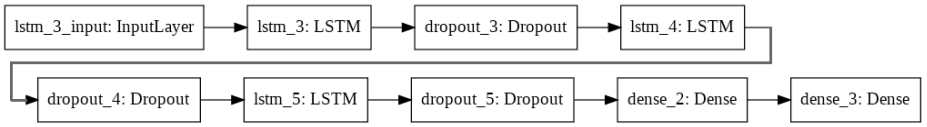


Fig. 2. Structure of Content Model.

As part of the conducted research, a couple of neural networks were designed and implemented. A chess game is a sequence of moves—hence the thesis that it is sequential data seems to be justified. One of the models tried to verify that assumption. The architecture of that model included LSTM recursive networks, which are designed to learn from sequential data. The neural network trained on the moves made by chess players has been named **Content Model**. In the rest of the paper, such a name will be used to denote this particular network.

The next model is designed to achieve the best possible results in terms of classification accuracy. It combined the network model discussed above and the neural network that learned on chess game metadata. These two combined models then form a classifier, able to predict whether the game is currently won by black, white, or is it a draw. The model was named **Context Match Fusion** (CMF) due to the combined use of the two networks. Such a name will be used to describe that model in the rest of the paper. The general diagram showing the CFM model is presented in Fig. 1.

2.1. Content Model

The Content Model consists of recursive LSTM layers. With its help, we tried to answer the question of whether the moves made during a chess game can be treated as sequential data. LSTM cells are designed to model information that is sequenced, so it seems like a good idea to use this kind of neural network to verify the assumption. The diagram of the proposed network is shown in Fig. 2.

The exact structure of the model layers presents as follows:

- Input layer with input size (sequence_length): 768.
- LSTM—number of cells: 512, *tanh* activation function, dropout: 0.2.
- LSTM—number of cells: 256, *tanh* activation function, dropout: 0.2.
- LSTM—number of cells: 128, *tanh* activation function, dropout: 0.2.
- Dense layer—number of cells: 32, *relu* activation function.
- Classifying layer—number of cells: 3, activation function $softmax$.

Additionally, the *Adam* optimizer was used with the following parameters: learning_rate = 0.000001, beta_1 = 0.9, beta_2 = 0.99, asmgrad = False.

Two methods were used to prevent overfitting/underfitting of the model. First, the *Early Stopping* mechanism was used, which goal was to monitor the value of val_loss. When it increased with training the network (which means that

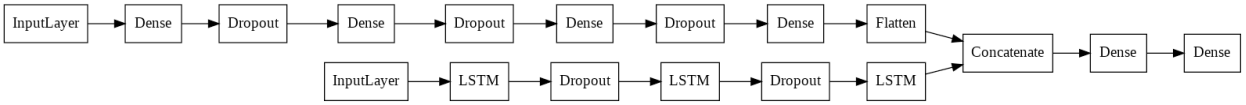


Fig. 3. Structure of model Context Match Fusion.

the model becomes worse) and such changes were observed for 5 epochs in a row, the training process was stopped. The next method was the *Model Checkpoint* mechanism. It saves the model as soon as it gets a better result on the validation set (which is computed after each epoch).

2.2. Context Match Fusion

The second proposed model is a one that combines the model presented above (responsible for processing the record of moves played in a chess game) and the part responsible for processing the data describing the chess match. Its structure is shown in Fig. 3

The exact structure is as follows—let's start with the DBN network:

- Input layer with input size: 4.
- Dense layer—number of cells: 4, activation function *relu*, dropout: 0.2.
- Dense layer—number of cells: 16, activation function *relu*, dropout: 0.2.
- Dense layer—number of cells: 12, activation function *relu*, dropout: 0.2.
- Dense layer—number of cells: 8, *relu* activation function, dropout: 0.2.

The architecture of the network responsible for processing the chess game moves is exactly the same as in the Content Model. Then the outputs of both of these networks are concatenated and a classifier is build consisting of:

- The layer connecting the outputs from the DBN and LSTM networks.
- Dense layer—number of cells: 64, *relu* activation function.
- Dense layer—number of cells: 3, $softmax$ activation function.

The part responsible for processing information describing the game, i.e. the ELO rankings of both players involved in the game and their names and surnames, is a network consisting only of dense layers. This architecture was chosen due to the nature of the data, which are four numerical data (the names of the players have been categorized).

In the experiments where the CMF model was used, we set up *Adam* optimizer with the following parameters: learning_rate = 0.001, beta_1 = 0.9, beta_2 = 0.990, asmgrad = False. Also, the methods preventing the model from overfitting (*Early Stopping* and *Model Checkpoint*) were used. Both methods were discussed in Section 2.1.

2.3. Dataset

The project uses a dataset that contains records of chess games in PGN (Portable Game Notation) format. It is the most popular format of this type, commonly used by databases storing game records. It contains tags encoding particular information about a chess game and a record of the chess game—sample file structure is presented in Fig. 4.

Data from the KingBase-Chess [7] portal were used in the study. This database contains the total number of 2185555 records of chess games including 799236 games with a victory for white, 606831 games with a victory for black, 779325 games with a draw and 163 games with a different result (e.g., a game in progress). All results include players with an ELO rating greater than 2000 and cover the time frame from 1990 to 2018.

However, this amount of data turned out to be too large for this research. The number of experiments when it was necessary to train the neural network forced the limitation of the dataset—we use only games from 2017 to 2018. The data has been divided into three parts: training set (including 120719 games from 2018), validation set (including 22413 games from 2017) and test set (including 22796 games from 2017).

When analyzing the particular sets, the data present as follows:

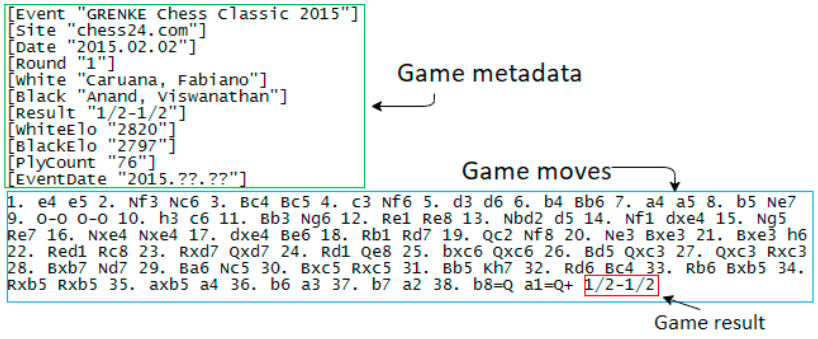


Fig. 4. Sample PGN file format.

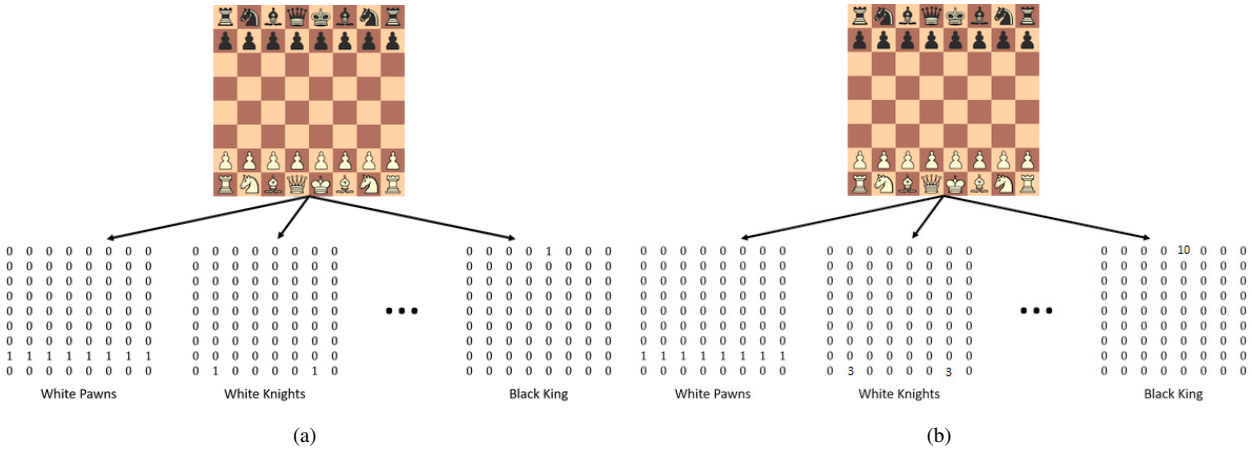


Fig. 5. (a) Bitmap and (b) Algebraic board representations.

- Training set contains 44196 games won by white, 34000 games won by black and 42523 draws.
- Validation set contains 8299 games won by white, 6950 games won by black and 7164 draws.
- Test set contains 8396 games won by white, 6858 games won by black and 7542 draws.

It is worth notice, that the sets were divided in such manner, that around 60% of the entire dataset was designated for the training set, 20% for the validation set, and 20% for the test set. Additionally, the sets are balanced according to the results of the game, which is a crucial aspect of this research because the designed classifiers tried to predict the result of a chess game.

2.4. Chessboard Representation

An important aspect of the research carried out in this paper is the influence of the type of vector representation of the chessboard on the accuracy of recursive LSTM networks learning. We focused on two different possibilities of conversing a chess position to a form that allows for training neural networks. They will be discussed below.

The first type of board representation is *bitmap representation* [11] which is presented in Fig. 5(a). In this method, each of 64 squares of the chessboard is represented as a binary vector of length 12. Each element of this vector is responsible for representing one particular piece and the side it belongs (black or white). A vector consists of 12 elements because that is exactly the number of different pieces on the board—6 pieces for blacks and 6 pieces for whites. Such a representation of the board gives the total length of the vector equal to 768, which reflects the full chess position.

The second type of board representation considered in this paper is *algebraic input* [9] — its main idea is presented in Fig. 5(b). This method makes a slight difference to the *bitmap input*. It not only takes into account whether a piece is present on the chessboard but also perceives a different value of pieces (for example a pawn is less worthy on the chessboard than a queen). Pawns are described as 1, knights and bishops as 3, rooks as 5, queens as 9, and kings as 10. It can be concluded that this method carries more information—not only piece position but also its value. The experiments conducted in this paper tried to verify whether this change makes a real improvement in the model learning process.

2.5. Data Preparation and Training Process

Content Model and one part of the Context Match Fusion model were designed to be trained on data containing moves made during a chess game. At this point, it is worth noting that a chess match may vary in length. The game can end after a few moves when one of the players makes a mistake at the beginning. On the other hand, it is not uncommon for the game to last for more than 100 moves. As a curiosity, the following records can be given: the shortest tournament game played by chess masters lasted only 3 moves; the longest tournament game was played in Belgrade in 1989—Ivan Nikolic and Goran Arsovic played a game of 269 moves, it lasted more than 20 hours and ended in a draw.

So, the games in the prepared dataset may have different sequence lengths. As an input to the model, we need a fixed-length sequence. Therefore, longer or shorter data must be either truncated or extended to a given size. For example, when a game has 50 moves, and we assumed that our model will be trained with a sequence of 40 moves, we cut the last 10 moves. When a game has a duration of 30 moves, and we have assumed that our model will be trained with a sequence of 40 moves, we need to extend the sequence by 10 moves. In our case, we extend it by adding "artificial" moves, that is vectors filled with zeros. In our experiments, we operated on three sequence lengths: 20, 50, 80.

The Context Match Fusion model contains two networks: the first processes the game flow, the second its context, i.e. the names of the players and their rankings. The problems arising from the first model have been discussed above. The second part of the model, a network composed of dense layers, also required some data preparation. Players' names and surnames have been categorized, which means that each individual player has been assigned an index. ELO rankings and categorized players have been normalized to a fixed range—in our case to (0, 1).

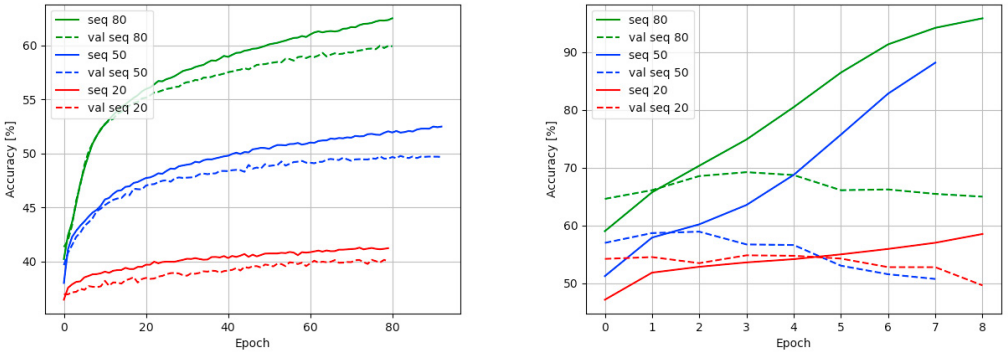
3. Results

This section presents the results that were achieved by performing the experiments solving the classification problem examined in this paper. The results will be split according to the board representation—first, we will introduce outcomes for bitmap input, next for algebraic input, and in the end, all the results will be compared.

3.1. Experiments Related to Bitmap Input

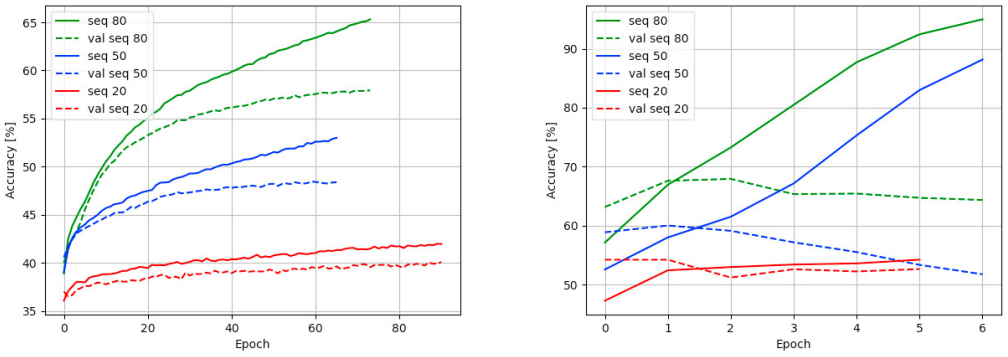
The diagram in Fig. 6(a) presents the relation between the accuracy and validation accuracy of the Content Model classification and the epoch of training. Each of the graphs shows one of three different sequence lengths—20, 50, 80. They have been compiled on one coordinate system to illustrate the relationship between the sequence size and the precision of model classification. The obvious conclusion is that the accuracy increased when the network was trained on the longer game sequence. The best results were obtained for games of length 80 (it is 80 plies, i.e. 40 moves in fact)—in such a case the accuracy of 65.32% was achieved, which is a very good result. For sequences of length 50 in recent epochs, the classification accuracy of 53% was achieved, and for sequences of length 20, the accuracy was 41.96

Next diagram presented in Fig. 6(b) shows results achieved for Context Match Fusion model. The experiments, of course, included training the model for three sequence lengths—20, 50, 80. An interesting point that can be observed when looking at the results for all sequence lengths is that the model overfitting occurs rapidly. The prediction accuracy and validation accuracy diverge already at an early stage of the network training process. However, despite that, the results observed on the validation set are still surprisingly high, and as it turned out, the same high percentage accuracy score also occurred when examining the model on the test set.



(a) (b)

Fig. 6. (a) Content Model and (b) Context Match Fusion classification and validation accuracies for different sequences lengths and bitmap input.



(a) (b)

Fig. 7. (a) Content Model and (b) Context Match Fusion classification and validation accuracies for different sequences lengths and algebraic input.

3.2. *Experiments Related to Algebraic Input*

Fig. 7(a) compiles prediction accuracy plots for the Content Model in one coordinate system—for a game of 20, 50, and 80 plies. As for the bitmap representation, for the algebraic representation, the best results were obtained for the longest sequence. We managed to achieve an accuracy of 62.51% for the 80 plies sequence, 52.49% for 50 plies sequence, and 41.23% for 20 plies sequence. These values were calculated based on the training set, however, it can be concluded that their high level is satisfactory.

As in the case of experiments using bitmap representation, the *Early Stopping* mechanism worked for models trained on the games of length 50 and 80. The mechanism interrupted the learning process because the accuracy of predictions on the validation set did not change, and on the training set it was constantly increasing. This can be observed in Fig. 7(a) where the graphs showing the accuracy on the training and validation set start to diverge with each epoch.

The results for Context Match Fusion model are presented in Fig. 7(b). As for the bitmap chessboard representation, also in this case the overfitting effect occurs quickly. Already during the first epochs of training, the models achieved the maximum accuracy of the classification on the validation set, and then only the result of classification accuracy on the training set was improved. Also, as in the case of the trained models based on the bitmap representation, the results achieved on the test set were high.

3.3. Comparison of Bitmap and Algebraic Input

In this section, all accuracies calculated on the test dataset, obtained from all experiments are put together for comparison. The obtained results presented in Table 1 will allow us to see possible discrepancies in the results depending on the representation.

Table 1. Comparison of the obtained results.

Model	Sequence	Input	Accuracy [%]
Content Model	20	Bitmap	38.39
		Algebraic	39.38
	50	Bitmap	49.68
		Algebraic	47.95
	80	Bitmap	59.62
		Algebraic	57.73
Context Match Fusion	20	Bitmap	34.57
		Algebraic	51.96
	50	Bitmap	51.14
		Algebraic	51.14
	80	Bitmap	68.9
		Algebraic	67.03

From the theoretical point of view, the algebraic representation carries more information, because, in addition to the data on the presence of a chess piece in a given field, it also contains data on the value of that piece. It could be assumed that for this very reason, the results should be better than those for the bitmap representation. Empirical results, however, showed something quite different. Higher levels of precision were achieved for the bitmap representation—only for small sizes of the sequence the algebraic representation had an advantage. The result achieved by the Context Match Fusion model with an algebraic representation for the sequence of length 20 is also puzzling—it is undoubtedly outlier in comparison to the analogous model trained on the bitmap representation. This may indicate some instability of the Context Match Fusion model.

4. Discussion

Our research goals included conducting a series of studies to determine if it is a good idea to use recursive LSTM neural networks to model a game as well structured as chess. In addition, a neural network model was proposed, combining two types of data: the moves played in the game and its context, i.e. who participates in the game on both sides, and what are the ELO rankings of these players. This model was aimed at achieving the best possible results in the classification of the games into one of three categories—white victory, black victory or draw. Additionally, two possible representations of the chessboard in the form of vector were verified (it was discussed in Section 2).

A large part of the effort put into the development of this research was devoted to the preparation of an appropriate data set. The challenge was to choose a suitable library containing historical records of chess games. Then, due to the enormous number of data, a reasonable subset had to be selected. Thus, over 120000 games from 2018 were allocated to the training set, almost 22000 games from 2017 to the validation set, and the same amount of games from 2017 to the testing set.

The experiments conducted with the Content Model were aimed at answering the question of whether the moves played during the chess games can be successfully treated as sequential data. The results of these experiments are described in Section 3. The research was carried out on three sequence lengths: 20, 50, and 80 plies. The results at the level of 59.62% accuracy for the bitmap representation and 57.73% for the algebraic representation (for a sequence length of 80) are satisfactory. Therefore, it can be assumed that the record of moves made by chess players is a kind of sequential data.

The Context Match Fusion model aimed to obtain the best possible results in the problem of classifying a chess game into one of three classes—white win, black win or draw. The obtained results turned out to be good—68.9% prediction accuracy for the bitmap representation and 67.03% for the algebraic representation. It is worth mentioning that randomly selecting one category out of three would give us a result of 33.3%, so, we managed to get a result that

is more than twice as good. It can, therefore, be considered a good idea to design a "fusion" of two networks that process two different types of data.

Another goal of the research was to compare the two ways (bitmap and algebraic) of representing the chessboard in the form of a vector. The results turned out to be surprising because, despite the assumption that the algebraic representation (theoretically carrying more information—about the position of a piece and its value) could achieve better accuracy, the tested models showed something completely different. Bitmap representation turned out to be a more effective tool, which may be useful knowledge for other research works on similar topics.

5. Conclusions

In this paper, we proposed a chess game prediction system using deep learning. We designed a couple of chess models that can predict a result of a game (white victory, black victory, or draw) while it is still in progress. We have applied the proposed approaches on a large dataset prepared by ourselves and consisting of chess matches records from 2017 and 2018.

We also posed a couple of research questions and answered them based on the results of our experiments—for the summary of our research findings please see Section 4. Some of the obtained research results—for example those on the different representations of the chessboard in the form of a vector—can be the guidance for the researchers working on similar problems. Also, our Context Match Fusion model received a high prediction accuracy, sometimes competitive and sometimes comparable with the results discussed in Section 1.

There are plenty of ideas on how future works, which are related to the research described in this paper, may be conducted. It would be interesting to figure out how Convolutional Neural Networks (CNN) are capable to handle similar problems as the examined ones—this could address the question of whether chess positions could be interpreted as spatial data. Secondly, it might be interesting to verify whether a similar approach can be used for other types of games, for example card or strategy games [12, 2].

Acknowledgements

This research was partially supported by the funds assigned to AGH UST by the Polish Ministry of Science and Higher Education.

References

[1] Doggers, P., 2019. World chess championship 2018: Carlsen vs Caruana. https://tinyurl.com/snanvcqu.
[2] Dreżewski, R., Klęczar, M., 2018. Artificial intelligence techniques for the Puerto Rico strategy game, in: Jezic, G., Kusek, M., Chen-Burger, Y.H.J., Howlett, R.J., Jain, L.C. (Eds.), Agent and Multi-Agent Systems: Technology and Applications. 11th KES International Conference, KES-AMSTA 2017 Vilamoura, Algarve, Portugal, June 2017 Proceedings, Springer International Publishing, Cham, Switzerland. pp. 77–87.
[3] Fan, Z., Kuang, Y., Lin, X., 2013. Chess game result prediction system. Machine Learning Project Report CS 229. Stanford University. Stanford, CA.
[4] Ferreira, D.R., 2010. Predicting the outcome of chess games based on historical data. Technical Report. IST—Technical University of Lisbon). Porto Salvo, Portugal.
[5] Ferreira, D.R., 2012. Determining the strength of chess players based on actual play. ICGA journal 35, 3–19.
[6] Hochreiter, S., Schmidhuber, J., 1997. Long short-term memory. Neural Computation 9, 1735–1780.
[7] KingBase Chess, 08.2020. Chess database. https://www.kingbase-chess.net/.
[8] Masud, M.M., Al-Shehhi, A., Al-Shamsi, E., Al-Hassani, S., Al-Hamoudi, A., Khan, L., 2015. Online prediction of chess match result, in: Advances in Knowledge Discovery and Data Mining. PAKDD 2015, Springer, Cham, Switzerland. pp. 525–537.
[9] Sabatelli, M., 2017. Learning to Play Chess with Minimal Lookahead and Deep Value Neural Networks. Master's thesis. University of Groningen, Faculty of Mathematics and Natural Sciences. Groningen, Netherlands.
[10] Sala, G., Foley, J.P., Gobet, F., 2017. The Effects of Chess Instruction on Pupils' Cognitive and Academic Skills: State of the Art and Theoretical Challenges. Frontiers in Psychology 8.
[11] Thompson, K., 1996. 6-piece endgames. ICGA Journal 19, 215–226.
[12] Wilisowski, Ł., Dreżewski, R., 2015. The application of co-evolutionary genetic programming and TD(1) reinforcement learning in large-scale strategy game VCMI, in: Jezic, G., Howlett, R.J., Jain, L.C. (Eds.), Agent and Multi-Agent Systems: Technologies and Applications. 9th KES International Conference, KES-AMSTA 2015 Sorrento, Italy, June 2015, Proceedings, Springer International Publishing, Cham, Switzerland. pp. 81–93.