2023 27th International Conference on Engineering of Complex Computer Systems (ICECCS) | 979-8-3503-4004-4/23/$31.00 ©2023 IEEE | DOI: 10.1109/ICECCS59891.2023.00011

Sports Analytics Using Probabilistic Model Checking and Deep Learning

Jin Song Dong*, Kan Jiang*, Zhaoyu Liu*, Chen Dong†, Zhe Hou‡, Rajdeep Singh Hundal*, Jingyu Guo*, and Yun Lin§
National University of Singapore*,Louisiana State University†,Griffith University‡,Shanghai Jiao Tong University§

***Abstract*—Sports analytics encompasses the use of data science, AI, psychology, and IoT devices to improve sports performance, strategy, and decision-making. It involves collecting, processing, and interpreting data from various sources such as video recordings and scouting reports. The data is used to evaluate player and team performance, prevent injuries, and help coaches make informed decisions in game and training. We adopt Probabilistic Model Checking (PMC), a method commonly used in reliability analysis for complex safety systems, and explain how this method can be applied to sports strategy analytics to increase the chance of winning by taking into account the reliability of a player's specific sub-skill sets. This paper describes how we have integrated PMC, machine learning, and computer vision to develop a new and complex system for sports strategy analytics. Finally, we discuss the vision of a new series of international sports analytics conferences (https://formal-analysis.com/isace/2023/).**

***Index Terms*—sports analytics, probabilistic model checking, object tracking, action recognition, deep learning**

I. Introduction

The motivation of sports analytics is to use quantitative data analysis and modeling techniques to better understand the complex decision-making processes that occur during competitive sports games. By analyzing historical data and using advanced modeling techniques, sports analysts can identify patterns and trends that can inform strategic decision-making and help players and coaches improve their chances of winning.

In professional sports, intelligent decision-making is crucial for increasing the winning chances. For instance, in tennis, players must determine where to serve, return, and place each shot to take advantage of their strengths and exploit their opponent's weaknesses. Randomizing decisions is also necessary to deceive the opponent, such as using a 60% wide, 30% T, and 10% body serve distribution. Quantifying player skill levels through shot success rates is important. Our aim is to understand the relationship among winning chance, decisions, and skill levels.

This is similar to the reliability analysis in a complex system. For example, the reliability of an aircraft can be calculated based on the reliability of the inter-connected components, such as engines, wings, sensors, etc. This similarity leads to the idea, which is the use of Probabilistic Model Checking (PMC) to compute the winning chances based on the sequence of decisions and the related success rates. We are the first to have applied PMC into tennis analytics [1] and soccer analytics [2].

We build the PMC game model using historical data to infer play patterns and skill levels of all players. Manual data collection by watching recorded videos was time-consuming and error-prone and limited to professional tournaments. To help a wider range of players, we propose deep learning-based models to automatically process YouTube broadcast videos. Our multi-mode learning approach includes text boxes, sound, and images to track balls and identify actions, which is challenging with low-resolution video.

When applying our PMC method to team sports strategy analysis, there is a "state explosion" problem because the number of states grows exponentially with respect to the number of players. The commonly used Value Iteration algorithm becomes computationally prohibitive because it is impossible to store all state values exactly. We propose that it is possible to store the approximated probabilistic reachability function in a linear regressor compactly. Furthermore, we propose to use tree search to improve (to reduce) the approximation error.

II. Method

The prerequisite of sports analytics is the collection of detailed match data, as illustrated in Figure 1. Therefore, we gather data from various websites and employ a video analytics program to automate the data collection process. Once the data is collected, we utilize machine learning tools to identify injury patterns, and use model checkers to quantitatively analyze how a player's play pattern and sub-skill level impact their overall winning chances. To facilitate team sports analysis, we have improved the model checker to enable the execution of models with a large number of states. These modules together comprise a comprehensive system. In the subsequent sections, we elaborate on each module in detail.

A. Court detection

The goal of court detection is to find a projective transformation matrix [3] which projects a standard court to the one visible in the video frame. The inverse of the transformation matrix allows the pixels in the video frame to be projected back to the standard court, hence, provides measurements in real-world coordinates.

Although the final goal is to find the transformation between the standard court and the one in the image, it is actually better to use two real images, as shown in Figure 2, because the standard court is visually very different from the actual

979-8-3503-4004-4/23/$31.00 ©2023 IEEE
DOI 10.1109/ICECCS59891.2023.00011

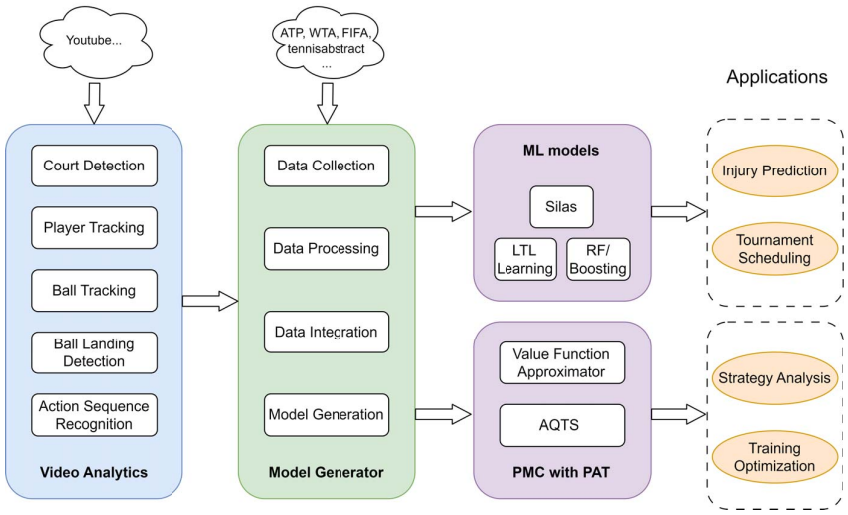

Fig. 1. Sports Analytics Overview. Video Analytics are deep learning based programs to automatically infer player's success rates of various actions. Model generator creates model based on historical data and domain knowledge. PAT is a Model Checker, which can automatically calculate the winning chance.

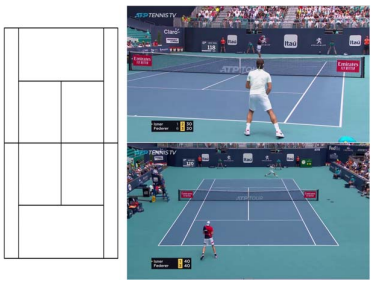

Fig. 2. Standard(left), source(top) and reference(bottom) court images

image. One important observation is that the difference between the two real images is not solely caused by the court transformation. Anything above the court plane, for example, the net, players and the audience do not follow the court transformation. To exclude those pixels, we need a mask function, which separates court line pixels from other pixels.

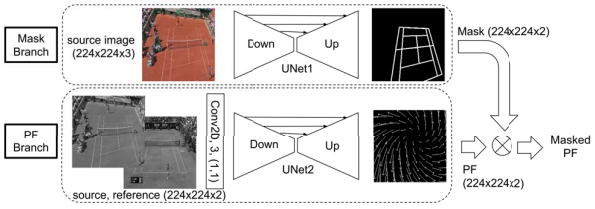

Fig. 3. Masked PF network architecture.

As shown in Figure 3, our solution consists of two branches. The mask branch detects the court line pixels. For the second branch (PF Branch), the input is a pair of grayscale images, where one is the source image and the other is a reference image. The output is the pixel-wise transformation between the two images, also known as Perspective Fields (PF). Multiplying with the mask, we obtain the masked PF, representing the court line pixel transformation.

B. Player and ball tracking

For player and ball tracking sub-tasks, the main challenges are missing detection and false detection. Since the ball is small, it is often occluded by players, the racket, or the net. Also, the ball can move so fast that it appears to be a thin blurring line. In soccer videos, players are often occluded by other players in front of them due to the camera angle. All these are the common reasons for missing detection. False detection mostly occurs because there are many small and ball-like objects or because several players are stacked together. To overcome these challenges, we propose a method of combining Convolutional Neural Network (CNN) [4] and Bayesian Estimations [5] to detect and track the objects by utilizing both spatial and temporal feature correlations. The CNN is trained to detect the correct object in the spatial context, e.g., a ball is detected not only because it has a ball-like shape and colour but also because it appears in the correct part of the image. The Bayesian estimation utilizes temporal correlations; that is, the ball and player position in the previous frame highly correlates to the position in the current frame. Using this temporal correlation, we can filter out false detection and fill in the missing detection. Once the ball trajectory is detected, we then train a Random Forest classifier to identify the ball landing positions (bouncing points).

C. Action sequence recognition

To extract a sequence of shots, such as serve, forehand or backhand shots, from the long video, we first locate each hitting moment in the time axis. This is achieved by slicing the entire match video into many short clips, each is 0.72 seconds long, and then classifying whether it contains a hitting. As shown in Figure 4, we use audio data as the main guiding

feature for hit detection, and we use the video to provide the visual clues as the assisting feature. Intuitively, this matches with the natural human experience of combining hearing and vision at the same time to identify the hitting events.

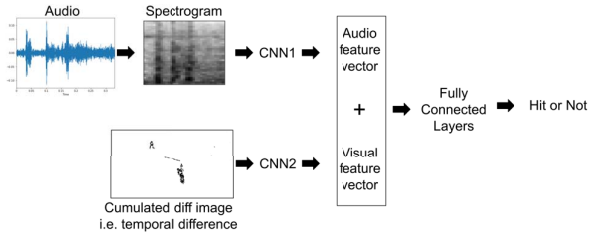

Fig. 4. Hitting detection DNN [6] uses both audio and visual features to improve detection accuracy.

To zoom into a small region where the action takes place, we utilize our player tracking result, because hitting can only happen near one of the players. After cropping according to the player bounding boxes, we then retrain the MMAction2 [7] toolbox (an open-source toolbox for video understanding) to classify the type of shots.

D. Model generator

To generate the game model, we begin by collecting data from relevant sources, such as tennisabstract.com, ATP, WTA and FIFA websites. Since some data are manually annotated by fans, it is essential to eliminate any incorrect data. Finally, by merging data from both the online data sources and the video analytics system, we have compiled tennis data for the past 12 years (2011-2022), and English Premier League for the past 6 years (2016-2021). The tennis data comprises 8,076 ATP and WTA matches with detailed shot-by-shot information, which involves 1,073 players and a total of 6,036,382 actions.

Creating models for tennis matches is a challenging process that requires a balance between accuracy and efficiency. The perfect model should capture the dynamic conditions of the game while maintaining the appropriate level of abstraction for explainability and efficient analysis. Our modeling approach is designed to address these requirements by providing an expressive representation and powerful analytical capabilities.

In a tennis single's match, the two players are commonly referred to as $P1$ and $P2$. To predict the match outcome, we analyse the winning probability in a **tiebreak game** model, which is an abstraction of the entire match. In a tiebreak game, the first player to score 7 points is declared the winner. We assume that the player with the highest probability of winning the tiebreak game is also more likely to win the entire match. This abstraction facilitates efficient performance verification using a model checker.

Our model is event-based. For example, each point begins with a service event. It is followed by a second serve event if the first one fails. Else, depending on the landing position and the player's handedness (whether the player is a right-handed or left-handed), the next event can be either a forehand return or a backhand return. Depending on the return direction, the next event can be a deuce court, middle court or ad court stroke. If the return fails, a point is awarded to the server. This event-based, story-like model is designed to be easy to understand.

The transition probabilities from one event to another are counted based on historical matches between $P1$ and $P2$, or similar players (such as players with the same handedness and similar Elo ranking points). The transition probabilities can also be systematically adjusted to simulate different strategies. Our tennis model focuses on two types of strategies. The first, called in-game strategy, involves detailed tactics about play patterns. For example, one can shift 10% T serves to W serves against a particular opponent who is weaker in returning W serves. The second type of strategy is training strategy, which aims to increase the success rate for a certain type of shot through targeted training. For instance, Federer may concentrate on practicing his backhand down-the-line shots before playing against Nadal, who is known for hitting powerful forehand shots to the opponent's backhand. This training not only increases Federer's backhand success rate but also reduces the threat of Nadal's forehand because the down-the-line shot forces Nadal to hit backhand shots.

We use the Probability Communicating Sequential Programs (PCSP) [8] to specify the event-based game model and PAT model checker [9] to calculate the winning probabilities.

E. Injury prediction

Each week there are 2-3 professional top-level tournaments around the world offering attractive prize money and ranking points for the players. Even though match playing is an important part of the professional player's career development, sports injuries will happen if the player plays too many tournaments. Based on historical data, when players retired in a match, we can trace their tournament schedule and performance, to identify a particular scheduling pattern that leads to a high risk of injury. The pattern is represented by a Linear Temporal Logic (LTL) formula built up from a set of Atomic Propositions (AP) and temporal operators. To improve the efficacy and recall of the learnt LTL, we use Silas [10], a high-performance machine learning tool with in-built logic reasoning and verification capability.

F. Scale up PMC

In team sports, it is common to see models with a very large number of states. For example, in a soccer game model, suppose we divide the soccer field into a 7x10 grid. Excluding the 2 gatekeepers, assume each of the remaining 20 players move within a certain region and assume each of these regions is a 3x4 grid, this is already 12^{20} different states.

The commonly used algorithm to calculate reachability in the PCSP model is the Value Iteration (VI) [11] algorithm. The value of a state represents the probability of reaching the winning state starting from the given initial state. When the number of states is very large, the VI algorithm becomes computationally prohibitive because: (1) It stores values of

all states in a long table, with one entry for each state. (2) It updates all state values at each iteration. We apply the Approximated Value Iteration (AVI) technique to overcome the problem by: (1) Value for all states are **compactly** stored in a linear **regressor**. (2) Value for **small number of** randomly sampled states are updated, and are used to retrain the linear regressor at each iteration.

Through AVI, we obtain an approximately correct value function. To further reduce the approximation error, we add an Adaptive Q-Value Tree Search (AQTS) component. Figure 5 shows the value at the root node is back propagated from the leave nodes. Some of the leave nodes are final states whose true values are already known, while other leave nodes have approximately correct values. During the backpropagation, errors are reduced and cancelled. The error reduction is significant when the tree is deep enough. To control the size of the tree, we carefully trim the useless nodes if their value is lower than the sibling's values.

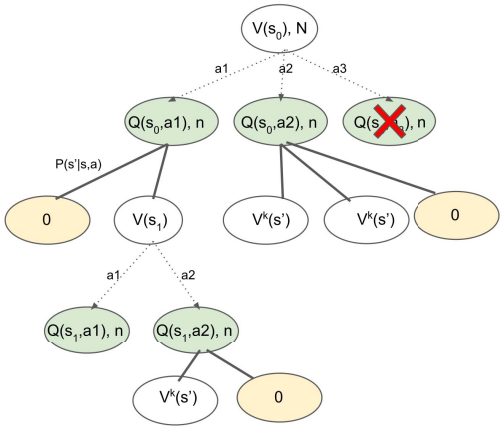

Fig. 5. Illustration of the Adaptive Q-value Tree Search. Final states are indicated as 0. Nodes with clearly low values compared with their siblings are trimmed (indicated as red X) to contain the size of the tree.

III. Results and Future Directions

To evaluate our model, we want to know: (1) How accurately can we predict the match outcome? (2) Does the strategy suggested by our model align with reality?

A good winning probability model should provide calibrated probability estimations. For example, if player $P1$ has a winning probability of 40% against $P2$, it means that if they play 100 times, $P1$ will win approximately 40 times. However, this is not possible because each match is played only once. Instead, we collect all matches with a predicted winning chance of 40% and see whether about 40% of the corresponding matches actually resulted in the player's winnings. Therefore, we evaluate the model's calibration using the expected calibration error (ECE) [12]. In addition to calibration, we also use Brier score (BS) [13] and log-loss (LL) [13] to quantify how close the forecasts are to the actual outcome.

Table I shows our model predicts the match outcome more accurately than many other machine learning models, such as Neural Network (NN), Support Vector Machine (SVM), Random Forest (RF), and CatBoost. When compared with bookmakers such as Bet365, our ECE is better while the BS and LL scores are slightly lower. This is likely because the bookmaker has information such as weather, recent forms, injuries, insider news, etc., which are hard to obtain.

TABLE I
The result of the experiment for win prediction.

	NN	SVM	RF	CatBoost	Ours	Bet365
ECE	0.048	0.009	0.030	0.042	**0.004**	0.016
BS	**0.240**	0.241	0.247	0.247	**0.240**	**0.214**
LL	0.673	0.675	0.694	0.689	**0.672**	**0.622**

We simulated betting with two strategies: (1) if our model predicts that a player has at least 60% probability of winning, we will bet on that player; (2) if our model predicts that a player has a winning probability that is higher than the bookmaker's prediction (intuitively, we think the bookmaker underestimates the player) by at least 8%, we will bet on that player. Table II shows that both strategies made long-term profits with annualized ROI at 3.92% and 5.32%, respectively.

TABLE II
Betting results over the past 10 years.

	Num of bets	Profits	ROI	Annualised ROI
Strategy 1	1,748	$4,693	46.93%	3.92%
Strategy 2	2,304	$6,795	67.95%	**5.32%**

It is challenging to determine the actual effectiveness of strategies suggested by our model because we cannot ask professional players to alter their playing patterns or skill levels and observe the impact on their win rates. Thus, we evaluate the effectiveness of these strategies using historical data. For instance, consider the 16 recorded matches between Roger Federer and Rafael Nadal. Before 2017, Federer only won 2 out of 10 matches. But after 2017, his win rate improved to 83%. Upon analyzing Federer's actions and sub-skill reliability, a notable difference is observed before and after 2017. To determine if this improvement is due to changes in strategy, we created two models based on historical data before and after 2017. The result shows Federer's winning chances were 35.7% and 53.2% respectively, which match the actual results. Table III summarises the empirical results, which show that the majority of the best actions identified by our system align with players' actual strategy adjustments and improvements. As such, we can conclude that our strategy analytics are reasonable and effective.

In soccer experiment, we use 6 seasons (2016-2021) of English Premier League matches (a total of 2280 matches) to predict winning chances using team formations and player ratings from EA's FIFA database. Our model achieves 43%

TABLE III
SYSTEM SUGGESTED ACTIONS FOR IMPROVEMENT COMPARED WITH PLAYERS' ACTUAL STRATEGY ADJUSTMENTS. "AG_I" DENOTES THE FRACTION OF OUR IN-GAME SUGGESTIONS THAT ALIGN WITH PLAYERS' ACTUAL STRATEGY ADJUSTMENTS. "AG_T" DENOTES THE FRACTION OF OUR TRAINING SUGGESTIONS THAT ALIGN WITH PLAYERS' ACTUAL SUB-SKILL IMPROVEMENTS.

Player vs opponent	Year	Win% before	Win% after	Ag_I	Ag_T
Federer vs Nadal	2017	20%	83%	9/11	8/11
Nadal vs Djokovic	2017	26%	60%	7/11	8/11
Murray vs Nadal	2015	14%	50%	10/11	8/11
Medvedev vs Zverev	2020	25%	83%	6/11	7/11
Zverev vs Tsitsipas	2021	20%	50%	8/11	7/11
Djokovic vs Tsitsipas	2020	50%	100%	9/11	8/11
Zverev vs Nadal	2020	20%	67%	6/11	8/11

accuracy for 3-class prediction (win/lose/draw) and 63% accuracy for 2-class prediction (win/not win). In simulated betting, we make 6.9% and 2.7% profits in the last two seasons, but losses in earlier seasons suggest inaccurate player ratings. Some wrong predictions result from "fake" formations submitted to deceive the opponent. (Table IV)

TABLE IV
SOCCER BETTING EXPERIMENT RESULT

Season	2015/16	16/17	17/18	18/19	19/20	20/21
Profit	-8.2%	-1.8%	-0.4%	-5.0%	6.9%	2.7%

In conclusion, sports analytics is a fast-growing field. According to the Mordor Intelligence report [14], the data-driven sports analytics market was valued at US$ 1.05 billion in 2020 and is expected to reach US$ 5.11 billion by 2026, growing at a Compound Annual Growth Rate (CAGR) of 30.13%. Besides its commercial value, there are many interesting and challenging problems waiting to be solved by the researchers. In our research group, there are multiple PhD students who devote their entire thesis [15] in this domain.

Currently, our MDP model is designed for a single agent, but we are exploring the integration of game theory into our system, as suggested by Fernando et al. [16], because most games are played by two players or two teams. We also plan to adopt some of the techniques used in AlphaGo [17], such as developing two competing models and learning optimal strategies by playing against each other. To support this effort, we have developed a nested model checker, N-PAT [18], which we intend to integrate into our sports analytics platform.

The accuracy of the deep learning modules is closely linked to the quantity and quality of the training samples. To address this, we are exploring the Active Learning approach proposed by Settles [19], which converts a large amount of unlabeled data into high-quality training data automatically. Another method of collecting a large amount of accurate data is through the use of smart IoT devices. A notable example of this is the successful partnership between the German soccer team and their technology partner SAP [20]. We are currently planning to collaborate with the Singapore National team in a similar manner.

REFERENCES

[1] J. S. Dong, L. Shi, K. Jiang, J. Sun *et al.*, "Sports strategy analytics using probabilistic reasoning," in *2015 20th International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE, 2015, pp. 182–185.

[2] S. Siddharth, S. Saurav, J. Kan, W. Bimlesh, and D. J. Song, "Model driven inputs to aid athlete's decision making," in *2020 27th Asia-Pacific Software Engineering Conference (APSEC)*. IEEE, 2020, pp. 485–489.

[3] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[4] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019.

[5] G. Bishop, G. Welch *et al.*, "An introduction to the kalman filter," *Proc of SIGGRAPH, Course*, vol. 8, no. 27599-23175, p. 41, 2001.

[6] K. Jiang, M. Izadi, Z. Liu, and J. S. Dong, "Deep learning application in broadcast tennis video annotation," in *2020 25th International Conference on Engineering of Complex Computer Systems (ICECCS)*. IEEE, 2020, pp. 53–62.

[7] M. Contributors, "Openmmlab's next generation video understanding toolbox and benchmark," 2020.

[8] J. Sun, S. Song, and Y. Liu, "Model checking hierarchical probabilistic systems," in *Formal Methods and Software Engineering - 12th International Conference on Formal Engineering Methods, ICFEM 2010, Shanghai, China, November 17-19, 2010. Proceedings*, ser. Lecture Notes in Computer Science, J. S. Dong and H. Zhu, Eds., vol. 6447. Springer, 2010, pp. 388–403.

[9] J. Sun, Y. Liu, J. S. Dong, and J. Pang, "Pat: Towards flexible verification under fairness," in *Computer Aided Verification: 21st International Conference, CAV 2009, Grenoble, France, June 26-July 2, 2009. Proceedings 21*. Springer, 2009, pp. 709–714.

[10] H. Bride, C.-H. Cai, J. Dong, J. S. Dong, Z. Hóu, S. Mirjalili, and J. Sun, "Silas: A high-performance machine learning foundation for logical reasoning and verification," *Expert Systems with Applications*, vol. 176, p. 114806, 2021.

[11] K. Chatterjee and T. A. Henzinger, "Value iteration," in *25 years of model checking*. Springer, 2008, pp. 107–138.

[12] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International conference on machine learning*. PMLR, 2017, pp. 1321–1330.

[13] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *the Journal of machine Learning research*, vol. 12, pp. 2825–2830, 2011.

[14] MordorIntelligence, "Sports analytics market - growth, trends, covid-19 impact, and forecasts (2022 - 2027)," https://www.mordorintelligence.com/industry-reports/sports-analytics-market, 2022.

[15] K. Jiang, "Sports strategy analytics using probabilistic model checking and machine learning," Ph.D. dissertation, National University of Singapore, 2023.

[16] D. Fernando, N. Dong, C. Jegourel, and J. S. Dong, "Verification of strong nash-equilibrium for probabilistic bar systems," in *International Conference on Formal Engineering Methods*. Springer, 2018, pp. 106–123.

[17] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[18] H. Bride, C.-H. Cai, J. S. Dong, R. Gore, Z. Hóu, B. Mahony, and J. McCarthy, "N-pat: A nested model-checker: (system description)," in *Automated Reasoning: 10th International Joint Conference, IJCAR 2020, Paris, France, July 1–4, 2020, Proceedings, Part II 10*. Springer, 2020, pp. 369–377.

[19] B. Settles, "Active learning literature survey," 2009.

[20] SAP, "Sap and the german football association turn big data into smart decisions to improve player performance at the world cup in brazil," https://news.sap.com/2014/06/sap-dfb-turn-big-data-smart-data-world-cup-brazil/, 2014.