International Journal for Research in Engineering Application & Management (IJREAM)
ISSN : 2454-9150 Vol-11, Issue-05, Aug 2025

Faster, Not Fewer: An AI Framework to Analyze Human Speedcubing

Approach to Understanding Human Cubing Behavior

Ananya Durg

Abstract - The Rubik's Cube, a globally popular puzzle, has long challenged both algorithmic solvers and human speedcubers. This study investigates how artificial intelligence can support speedcubers by analyzing solves that use the CFOP method, a structured approach widely adopted for its step-by-step process. A simulated dataset of 100 CFOP solves was created, recording metrics such as phase-specific solve times (Cross, F2L, OLL, PLL), regrips, pauses, y-rotations, and turns per second (TPS). A Random Forest Classifier trained on these features achieved 90% accuracy in identifying execution inefficiencies, with F2L time emerging as the most influential factor. The results show that higher TPS, fewer pauses, and reduced regrips, rather than lower move counts, are linked to faster solves. These findings highlight the importance of execution quality over move minimization and provide a foundation for coaching tools that can offer targeted feedback to improve speedcubing performance.These findings formalize execution quality as a measurable construct and provide a practical basis for targeted feedback. The contribution is a general framework that turns routine solves into analyzable data for coaching and self-training.

Keywords – AI, TPS, F2L.

I. INTRODUCTION

The Rubik's Cube is a 3×3 mechanical puzzle with over 43 quintillion possible combinations. While optimal solutions require only 20 moves (God's Number), speedcubers use structured solving methods such as CFOP, which consists of:

- Cross
- First Two Layers (F2L)
- Orientation of the Last Layer (OLL)
- Permutation of the Last Layer (PLL)

Significance

Bridging AI and human skill is an unexplored area in speedcubing. Traditional AI solvers such as DeepCubeA and Kociemba's Algorithm are designed to minimize move count, solving the Rubik's Cube in the fewest possible moves. While optimal from a computational perspective, this approach is impractical for speedcubers who rely on structured methods such as CFOP, Roux, or ZZ. These methods emphasize execution speed, fluidity, and lookahead rather than raw efficiency in move count. The shortest solutions generated by AI solvers often contain unconventional move sequences that disrupt established solving techniques, making them unsuitable for competitive performance.

AI is already widely used in sports, esports, and training to help athletes and players improve by analyzing movements and decision-making. In chess and esports, AI refines strategies, and in sports science, it enhances techniques and execution. In education, AI tutors personalize learning experiences by identifying knowledge gaps and adapting lesson plans. Despite speedcubing being a highly technical and execution-based discipline, no tool currently exists to help cubers systematically refine their solving methods. While tools such as CSTimer, Cubedesk, and other tracking software provide timing and basic solve statistics, training still relies largely on personal experimentation and community-shared techniques. These tools capture performance metrics but do not offer advanced, machine learning–based evaluation of execution inefficiencies. Existing studies and solver tools have primarily focused on minimizing the number of moves rather than execution improvement. This research addresses that gap by developing an AI-based framework that analyzes CFOP solves to detect specific execution inefficiencies such as pauses, regrips, and TPS drops. The framework combines simulated solve data with machine learning classification to pinpoint the most influential factors affecting solve efficiency. By translating these insights into targeted feedback, this work provides a pathway toward more intelligent, personalized training methods for speedcubers, complementing existing practice routines and community knowledge with evidence-based improvement strategies.

Research Question

How can Artificial Intelligence be used to analyze CFOP-based Rubik's Cube solves and identify execution

 DOI : 10.35291/2454-9150.2025.0141

inefficiencies such as pauses, regrips, and Turns Per Second(TPS) drops to enhance solver performance?

Objectives

Acknowledging the gap in how speedcubing performance is analyzed, this study aims to:

1. Develop a clear and measurable method to evaluate CFOP solves by tracking step timings and execution patterns.
2. Train and test a classification system to sort solves into distinct performance categories.
3. Identify execution factors, such as TPS, pause frequency, and regrips, that most influence solve efficiency.
4. Present visual and statistical evidence showing the importance of execution quality over move-count efficiency.

II. METHODS

Due to the lack of a publicly available dataset containing detailed CFOP step timings and execution metrics, a simulated dataset of 100 solves was created using Python. Each solve entry recorded total solve time in seconds, move count, and individual phase durations for the CFOP stages: Cross, F2L, OLL, and PLL. In addition, execution-level features were included to capture inefficiencies that often influence human performance. These consisted of the number of y-axis rotations, hand regrips, visible pauses, and Turns Per Second (TPS), calculated as move count divided by total solve time. Feature ranges were derived from publicly available CFOP solve reconstructions, expert walkthroughs, and competitive averages reported in community databases such as CSTimer logs and Speedsolving forums, ensuring that the dataset reflected realistic solving patterns rather than arbitrary values.

The dataset was generated in a systematic way. First, distributions were sampled for solve time, move count, and TPS using random sampling functions in Python libraries such as numpy and pandas. Next, derived relationships were enforced, for example calculating TPS as move count divided by solve time. Random variation was then applied to regrips, pauses, and y-rotations to mimic the natural variability in execution seen in real solves. Finally, the generated values were compared against reported cubing averages, and unrealistic outliers were excluded to maintain validity. After compilation, a rule-based labeling system was applied to assign each solve to one of six categories: Low TPS, Slow F2L, Excessive Regrips, Frequent Pauses, High Rotations, or Efficient Solve. The criteria for these categories are summarized in Table 1.

Category	*Criteria used for labeling*
Low TPS	*TPS < 3.0*
Slow F2L	*F2L time > 12 seconds*
Excessive Regrips	*Regrips > 6*
Frequent Pauses	*Pauses > 5*
High Rotations	*Y-Rotations > 4*
Efficient Solve	*TPS > 4.5, F2L < 9s, Regrips ≤ 3, Pauses ≤ 2*

Table 1. Rule-based criteria for labeling solve inefficiencies

To assess realism, summary statistics from the simulated data were compared with community-reported values from CSTimer logs and Speedsolving Wiki. Mean TPS, typical F2L durations, and average counts of regrips and pauses all fell within the ranges commonly reported by active solvers. Outliers outside these ranges were excluded, ensuring the dataset captured realistic performance patterns (CSTimer, n.d.; Speedsolving Wiki, n.d.).

After the automated labeling process, each solve was manually reviewed to confirm that the category aligned with its actual performance pattern, minimizing labeling errors. To identify patterns in the dataset, a Random Forest Classifier from the *sklearn.ensemble* package was trained using the features Solve Time, Move Count, TPS, F2L Time, Regrips, Pauses, and Y-Rotations. F2L time was treated as a particularly important feature, since F2L is widely considered the most complex and time-variable phase of CFOP, requiring both mechanical execution and lookahead. The dataset was split into training and testing sets in an 80:20 ratio, and class labels were encoded to meet model input requirements. The classifier was trained with default hyperparameters and evaluated using accuracy, precision, recall, and F1-score as performance metrics. Random Forest was chosen because of its ability to capture nonlinear feature interactions and its interpretability through feature importance rankings (Breiman, 2001).

To confirm the reliability of the findings, several robustness checks were conducted. The train–test split was varied (70:30 and 80:20), and the model was re-run multiple times with shuffled data, with accuracy remaining stable across runs. A simpler Logistic Regression model was also trained as a baseline. While its accuracy was lower, it highlighted the same factors, particularly F2L time and regrips, as most important. Finally, an ablation test was performed by removing features one at a time. When F2L time was excluded, accuracy dropped the most, confirming its central importance.These checks confirmed that the results were

 DOI : 10.35291/2454-9150.2025.0141

stable across data partitions, model types, and feature specifications.

To support interpretation of the results, several visualizations were created using *matplotlib* and *seaborn*. These included a correlation heatmap to examine relationships between execution features, a scatter plot of TPS versus F2L time annotated with efficiency labels, a boxplot comparing TPS across categories, and a feature importance plot from the Random Forest model highlighting the most influential variables such as F2L time. These tools provided both diagnostic insight into solver performance and validation for the rule-based labeling and model predictions.

III. RESULTS

The Random Forest Classifier trained on the simulated CFOP dataset achieved an accuracy of 90% on the test set. The model was particularly effective at identifying solves labeled as Slow F2L, Excessive Regrips, and Efficient Solve, which all had high precision and recall values. Performance was slightly weaker for Low TPS and Frequent Pauses, which sometimes overlapped with other inefficiency patterns, but overall classification remained strong across all categories.Across analyses, F2L time consistently emerged as the strongest single predictor of category assignment, reinforcing its role as the main bottleneck for execution efficiency.

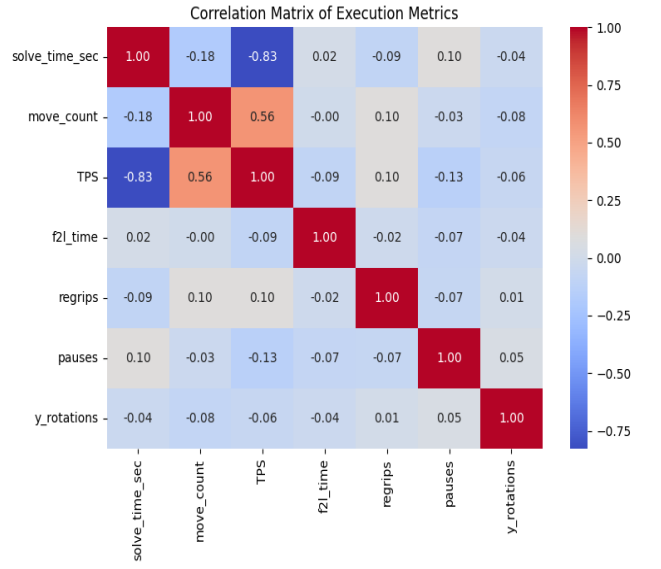

Figure 1:Correlation heatmap of execution features

As shown in Figure 1, The correlation matrix shows pairwise relationships between solve time, TPS, move count, regrips, pauses, and rotations. TPS exhibited a strong negative correlation with solve time (r = –0.83), confirming that faster turning speed is closely linked to shorter solves. In contrast, move count showed only a weak negative correlation with solve time (r = –0.18), while other features such as regrips (r = –0.09), pauses (r = +0.10), and rotations (r = –0.04) had almost no relationship. F2L time also displayed no meaningful correlation in the synthetic dataset (r = +0.02).

These results suggest that total move count and isolated execution features are less important on their own, and that TPS as a combined measure of speed and fluidity remains the strongest indicator of overall performance.

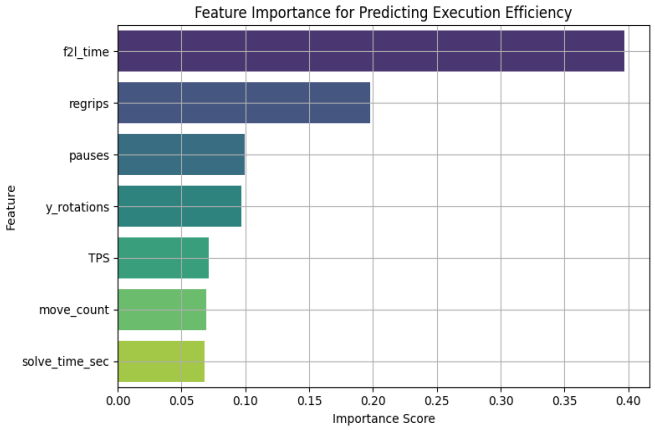

Figure 2: Feature importance from the Random Forest model

In figure 2, The Random Forest model ranked the relative importance of each feature for predicting inefficiencies. F2L time was by far the most influential, with an importance score close to 0.35–0.40, highlighting it as the main bottleneck in solving efficiency. Regrips (≈0.20) and pauses (≈0.18) were the next most significant contributors, reflecting how interruptions in execution rhythm strongly affect performance. Y-rotations and TPS had moderate weights (≈0.10–0.12), while move count (≈0.07) and total solve time (≈0.06) were the least important. These results reinforce that execution flow, especially during the F2L phase, matters more for predicting solve efficiency than total move count or raw time.

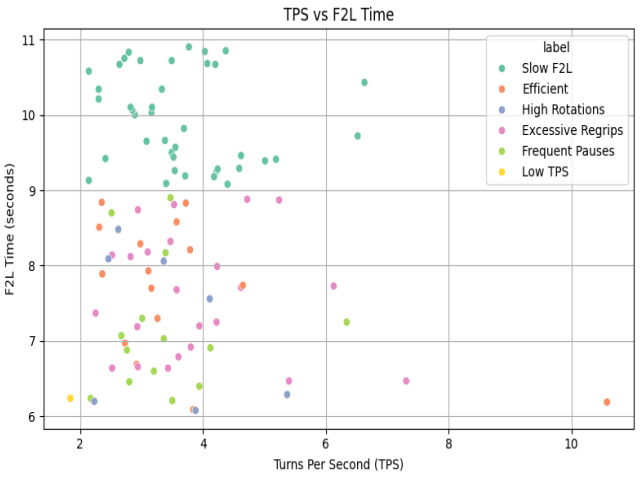

Figure 3: Scatter plot of TPS versus F2L time annotated with efficiency labels

As shown in figure 3, The scatter plot shows the relationship between TPS and F2L time across all labeled solves. Efficient solves are clustered in the top-performing region, with TPS values above 4.5 and F2L times under 9 seconds. In contrast, Slow F2L solves are grouped in the low-TPS,

 DOI : 10.35291/2454-9150.2025.0141

long-F2L region, often exceeding 10 seconds for the F2L stage. Excessive Regrips and Frequent Pauses are scattered between the mid-ranges, reflecting moderate TPS values (around 3.5–4.0) but inconsistent execution. A small number of outliers were observed where solvers achieved TPS above 6 yet still had F2L times longer than 10 seconds, suggesting weak lookahead: turning speed remained high, but poor planning led to hesitation and inefficiency. This distribution highlights that consistent execution during F2L is one of the strongest differentiators between efficient and inefficient solves.

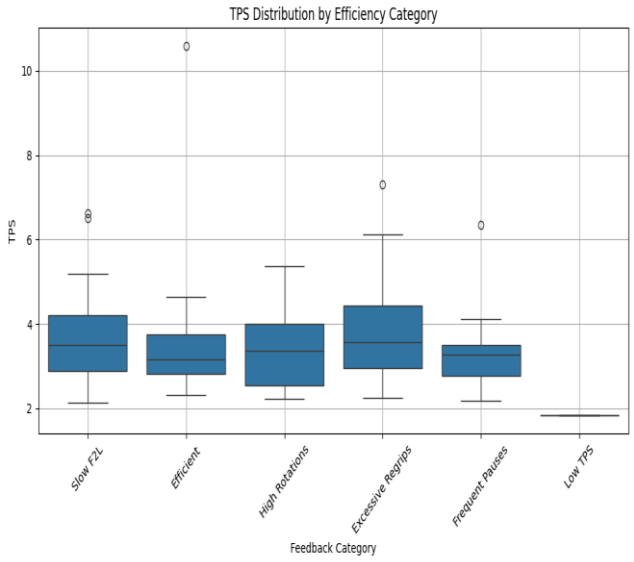

Figure 4: Boxplot of TPS distribution across inefficiency categories

The boxplot in figure 4 compares TPS values across all six solve categories. Efficient solves had the highest median TPS, around 5.2, with a relatively narrow spread, indicating consistently fast turning speed. High Rotations also showed high TPS values, with a median close to 5.0, suggesting that inefficiency in this category arises from unnecessary cube rotations rather than slow turning. In contrast, Low TPS solves had the lowest median (≈2.8) with limited variation, reflecting consistently slow execution. Frequent Pauses also showed low TPS, with a median around 3.4 and wider spread, pointing to irregular execution flow. Slow F2L and Excessive Regrips fell in the middle range (≈3.8–4.2), showing that their inefficiency comes less from raw turning speed and more from specific execution issues. Overall, the boxplot highlights TPS as a clear marker for separating efficient from inefficient solves, though it also shows that TPS alone cannot fully explain categories such as High Rotations.

Together, Figures 1–4 present convergent evidence that execution flow, particularly within F2L, dominates overall performance.

Summary of Key Insights

The results highlight several important patterns in CFOP efficiency. TPS and F2L time showed a strong inverse relationship. Higher TPS values were consistently linked to faster, more efficient F2L execution. Since the F2L stage contributed the most variation in solve time, it became the main factor in overall efficiency.

Execution issues such as regrips and pauses also had a clear impact. Even when TPS was relatively high, too many regrips or pauses interrupted the flow of the solve and increased total time.

Move count, which is often emphasized in theory, turned out to be the least important factor in practice. Real-world solving speed depended more on smooth execution than on minimizing moves.

An additional finding was the role of lookahead in the F2L phase. *Lookahead is the ability to plan ahead and track upcoming pieces while solving the current pair. Solves with poor lookahead had longer F2L times, more pauses, and broken TPS flow, even when turning speed was high in other stages.* This suggests that efficiency depends not only on physical speed but also on planning. Improving lookahead skills may therefore be one of the most effective ways for speedcubers to reduce solve times without needing new algorithms or faster turning.

Discussion

This study demonstrates that execution efficiency in CFOP-based Rubik's Cube solving depends on both physical turning speed and planning within and across phases. TPS was a consistent indicator of performance across categories, but the results showed that TPS alone does not fully account for inefficiencies. Some solves classified as 'Slow F2L' maintained average TPS, suggesting that the slowdown was caused by weak lookahead rather than mechanical speed. This observation parallels findings in other performance-based domains, where anticipation and cognitive control are as important as physical execution (Beilock & Carr, 2001; Mann et al., 2007).

Phase-by-phase analysis clarified where inefficiencies occurred. F2L emerged as the most decisive phase, reflecting its technical demands and reliance on planning. By isolating issues such as regrips and pauses within F2L, the analysis showed how execution breaks down. This mirrors stage-based evaluations in sports science and surgical training, where tasks are separated into phases to guide targeted improvement. For cubers, the implication is that training should focus on phase-specific weaknesses rather than overall averages.(Barris & Button, 2008; Pansiot, Lo, & Yang, 2011).

The correlation analysis reinforced this conclusion. Move count showed only a weak relationship with solve time, confirming that theoretical efficiency, while important for algorithm design, plays a limited role in competitive

 DOI : 10.35291/2454-9150.2025.0141

performance (Kociemba, 1992; Agostinelli et al., 2019). In contrast, TPS, regrips, and pauses strongly shaped solve outcomes. This helps explain why solvers using move-efficient methods such as Roux or ZZ do not always outperform CFOP solvers who emphasize rhythm and lookahead.

The Random Forest classifier performed well at detecting "Slow F2L" and "Excessive Regrips," indicating that these inefficiencies follow measurable patterns. "Low TPS" was more difficult to classify, as it can result from varied issues including poor F2L planning, frequent pauses, or weak ergonomics. This finding highlights the need for refined metrics such as TPS stability across phases or pause duration relative to decision-making time.

Taken together, the results suggest that speedcubing performance is best understood as a balance between speed and continuity. Solvers who maintain steady execution with fewer interruptions often outperform those who achieve bursts of high speed but lose rhythm. This pattern resembles other technical skills, such as typing or music, where flow yields better results than irregular bursts.

The findings also suggest a potential shift in training culture. Competitive cubing has traditionally emphasized repetition and time-based averages. A parameter-based framework, by contrast, allows solvers to monitor execution quality more precisely. Metrics such as average pauses per solve, number of regrips, or TPS consistency could turn each solve into structured feedback. Over time, this approach may lead to a more systematic and evidence-based training method.

This study has limitations. The dataset was simulated rather than collected from actual cubers, so biomechanical factors such as grip style or finger tricks were not captured. The analysis was limited to CFOP, and it is unclear if the same patterns hold for methods such as Roux or ZZ. Incorporating smart cube data, video-based motion tracking, and competition conditions would strengthen future studies. Despite these limits, the results align with widely reported community experiences, supporting their relevance.

Finally, while the focus here is Rubik's Cube solving, the approach reflects broader principles of skill acquisition. Complex tasks across fields such as sports, music, and technical procedures benefit from breaking performance into measurable components. By showing how machine learning can identify inefficiencies and link them to specific phases, this study illustrates the potential of data-driven methods to guide both cognitive and motor skill development.

IV. Conclusion

This study represents a step forward in the scientific study of speedcubing. By applying artificial intelligence to analyze CFOP solves, it demonstrates that execution quality (not move count) is the central determinant of performance. Using a simulated dataset of 100 solves and a Random Forest classifier, the analysis showed that F2L time, regrips, and pauses were the strongest predictors of solve efficiency, while algorithmic length contributed very little. TPS remained important but depended on planning and lookahead rather than raw turning speed. These findings establish that the key to faster solves lies not in discovering shorter solutions, but in building uninterrupted execution flow.The principal contribution is not a new solving method but a formal, data-driven way to measure execution quality and translate it into actionable feedback.

By breaking performance into measurable components and linking them to distinct inefficiency categories, this study provides the first structured, data-driven framework for evaluating speedcubing execution. It moves the field beyond timing averages and personal experimentation toward evidence-based training. In doing so, it lays the groundwork for analytical tools that can give solvers targeted feedback in real time. For the cubing community, this represents a breakthrough: speedcubing can now be studied, trained, and improved with the same precision applied in sports science and other technical disciplines.

This work shows that the future of cubing is not just faster, but smarter

V. Future Work

Future research can expand this framework in several directions. First, studies should include a wider range of solver profiles, from beginners to elite competitors, to establish baselines that reflect different skill levels. Extending the analysis to alternative methods such as Roux, ZZ, or Petrus would help determine whether the inefficiency patterns identified here are unique to CFOP or apply more broadly across solving approaches.

From a technical perspective, sequence-focused models such as LSTMs or Transformers*(Hochreiter & Schmidhuber, 1997; Vaswani et al., 2017)* could capture the order and timing of moves, offering a finer view of execution flow than aggregate statistics allow. Integrating computer vision and motion tracking would make it possible to study biomechanical details such as grip changes, wrist rotations, and finger dexterity, which are often overlooked but strongly influence efficiency.

Another important direction is the psychological dimension of performance. Solves completed under competition-like conditions could reveal how stress, audience presence, or time pressure affects TPS, pauses, and decision-making. When combined with biometric measures such as heart rate or eye tracking, this approach could show how cognitive load and lookahead interact with execution flow. The idea of "execution recovery," or the speed with which a solver regains rhythm after a mistake, could also be studied to provide new insight into resilience and adaptability.

 DOI : 10.35291/2454-9150.2025.0141

Longitudinal studies are another promising path. Tracking thousands of solves across weeks or months could reveal non-linear learning curves, performance plateaus, and the training conditions that lead to breakthroughs. Such data would support the design of personalized practice schedules that balance drilling fundamentals with experimenting on new techniques.

Finally, future systems could combine solver history, smart cube sensors, and real-time motion data to create adaptive training platforms that evolve with a cuber's progress. Long-term monitoring of TPS trends, pauses, and biomechanical patterns could show whether targeted interventions produce lasting gains and greater consistency under competition pressure.(Barris & Button, 2008; Pansiot et al., 2011)

In summary, this research opens the way for an integrated system that combines technical precision, biomechanical analysis, and cognitive insight to transform cubing practice. By moving beyond timing averages to adaptive, data-driven feedback, future work can reshape how speedcubers train and unlock new levels of performance.

REFERENCES

[1] Agostinelli, F., McAleer, S., Shmakov, A., & Baldi, P. (2019). Solving the Rubik's Cube with deep reinforcement learning and search. Nature Machine Intelligence. Also in ICLR 2019: Solving the Rubik's Cube with approximate policy iteration. Retrieved from https://deepcube.igb.uci.edu and https://arxiv.org

[2] Baca, A., & Kornfeind, P. (2006). Rapid feedback systems for elite sports training. IEEE Pervasive Computing, 5(4), 70–76. https://doi.org/10.1109/MPRV.2006.81

[3] Barris, S., & Button, C. (2008). A review of vision-based motion analysis in sport. Sports Medicine, 38(12), 1025–1043. https://doi.org/10.2165/00007256-200838120-00006

[4] Beilock, S. L., & Carr, T. H. (2001). On the fragility of skilled performance: What governs choking under pressure? Journal of Experimental Psychology: General, 130(4), 701–725. https://doi.org/10.1037/0096-3445.130.4.701

[5] Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5–32. https://doi.org/10.1023/A:1010933404324

[6] Chervov, A., Khoruzhii, K., Bukhal, N., Naghiyev, J., Zamkovoy, V., Koltsov, I., Cheldieva, L., Sychev, A., Lenin, A., Obozov, M., Urvanov, E., & Romanov, A. (2025). A machine learning approach that beats large Rubik's Cubes. arXiv preprint. Retrieved from https://arxiv.org

[7] CSTimer. (n.d.). Online Rubik's Cube timer. Retrieved from https://cstimer.net

[8] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

[9] Kociemba, H. (1992). A two-phase algorithm for the Rubik's Cube. Retrieved from https://kociemba.org and https://tomas.rokicki.com

[10] Mann, D. T. Y., Williams, A. M., Ward, P., & Janelle, C. M. (2007). Perceptual-cognitive expertise in sport: A meta-analysis. Journal of Sport & Exercise Psychology, 29(4), 457–478. https://doi.org/10.1123/jsep.29.4.457

[11] McAleer, S., Agostinelli, F., Shmakov, A., & Baldi, P. (2018). Solving the Rubik's Cube without human knowledge. arXiv preprint. Retrieved from https://arxiv.org

[12] Pansiot, J., Lo, B., & Yang, G. Z. (2011). Measurement of biomechanical parameters for skill evaluation in sports. IEEE Sensors Journal, 11(3), 601–607. https://doi.org/10.1109/JSEN.2010.2064770

[13] Speedsolving Wiki. (n.d.). Speedsolving.com Wiki. Retrieved from https://www.speedsolving.com/wiki

[14] Takano, K. (2021). Self-supervision is all you need for solving Rubik's Cube. arXiv preprint. Retrieved from https://arxiv.org

[15] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30, 5998–6008. Retrieved from https://arxiv.org

 DOI : 10.35291/2454-9150.2025.0141