Proceedings of the 20th Conference on Computer Science and Intelligence Systems (FedCSIS) pp. 825–830
DOI: 10.15439/2025F3675
ISSN 2300-5963 ACSIS, Vol. 43

Hybrid Boosting and Multi-Modal Fusion for Chess Puzzle Difficulty Prediction

Ming Liu
Jintai Stratech
China
lmymnew@gmail.com

Junye Wang
University of Science and Technology of China
China
wangjunye@mail.ustc.edu.cn

Yinghan Hu
Jilin University
China
h796431@qq.com

Xiaolin Yang
Inspur
China
yangxl@inspur.com

Defu Lin
Beijing Institute of Technology
China
lindf@bit.edu.cn

***Abstract*—The FedCSIS 2025 Challenge on Predicting Chess Puzzle Difficulty tasked participants with estimating puzzle ratings directly from board states and solution sequences, without relying on human solver statistics.**

We propose a three-stage hybrid framework integrating gradient-boosting regressors, a multi-modal neural network, and an XGBoost stacking ensemble. The boosting stage modeled handcrafted structural features derived from FEN and engine metadata, while the multi-modal network jointly learned from structured features and image-rendered chessboards to capture positional and tactical patterns. The residual-based stacking stage explicitly modeled prediction errors to correct systematic biases and enhance performance, particularly for high-difficulty puzzles.

Our method achieved a competitive performance, ranking 7th in the preliminary stage and 8th in the final leaderboard. These results demonstrate that combining interpretable boosting models with visual-tactical deep representations and meta-learning provides a robust and computationally efficient alternative to large-scale transformer-based approaches.

***Index Terms*—Chess puzzle difficulty prediction, Gradient boosting, Multi-modal learning, Deep learning, Residual-based stacking, Structural feature engineering, Uncertainty estimation**

I. Introduction

THE prediction of chess movements has evolved dramatically since the landmark achievement of IBM's Deep Blue in 1997[1], when it famously defeated world champion Garry Kasparov. Automated prediction of chess puzzle difficulty plays an increasingly important role in online training platforms, enabling adaptive puzzle recommendation and accurate tracking of player progression.

Early research focused on handcrafted features, employing gradient-boosting or support vector regressors trained on material balance, king safety, and piece mobility [2], [3]. Although these models lacked scalability, they demonstrated the importance of domain knowledge and interpretability in difficulty estimation. With the advent of deep learning, convolutional neural networks (CNNs) [4] were introduced to treat the chessboard as an image, successfully capturing spatial and tactical patterns such as attacking piece clusters and exposed kings. Later, hybrid CNN-LSTM [5] architectures integrated sequential move information, showing that temporal reasoning improves alignment with human-rated difficulty. Transfer learning approaches, such as DeepChess [6], adapted game prediction networks for puzzle rating, implicitly learning tactical patterns from large-scale game data.

More recently in the IEEE Big Data Cup 2024 [7], transformer-based architectures, such as GlickFormer [8], achieved state-of-the-art results by modeling spatio-temporal dependencies in move sequences. However, these models are computationally expensive and thus less practical for real-time puzzle rating systems. Competition-oriented solutions, such as the Bread Emoji team's hybrid ensemble combining engine-derived success probabilities with neural embeddings [9], have demonstrated that combining interpretable features with lightweight neural models can remain competitive while being computationally efficient.

Our team has an extensive history of successful participation in data science competitions hosted on the KnowledgePit platform[1]. We have consistently leveraged Gradient Boosting Decision Tree (GBDT) algorithms to tackle a wide range of predictive tasks—including classification, regression, forecasting, and image recognition—achieving top-ranked results and earning multiple awards[10] - [20]. Motivated by this strong background, we approach the present challenge with the same commitment to excellence.

Our work proposes a hybrid three-stage framework designed to achieve high predictive accuracy while maintaining interpretability and efficiency. We integrate 3 parts, which are

- three gradient-boosting models for robust tabular predictions,
- a multi-modal neural network to extract visual-tactical cues from rendered chessboard images,
- and an XGBoost stacking ensemble to fuse predictions into a single optimized output.

Additionally, we extend the method with a mask-based uncertainty estimation task, where the goal is to identify the most error-prone puzzles.

The remainder of this paper is organized as follows. Section II describes the challenge setup, dataset, and evaluation metric. Section III details the proposed methodology. Section IV presents the mask extension for uncertainty estimation. Section V reports experimental results and ablation studies. Finally, Section VI concludes the paper and discusses future research directions.

[1]https://knowledgepit.ai/

II. Challenge Description

The FedCSIS 2025 Challenge [21] is the second edition of IEEE BigData Cup 2024 chess puzzle competition [7] on Predicting Chess Puzzle Difficulty addressed the problem of estimating puzzle difficulty ratings directly from board configurations and solution sequences. The task was formulated as a regression problem, where each puzzle was assigned a continuous difficulty rating analogous to Lichess[2] puzzle ratings, typically ranging between 800 and 2800. These ratings approximate the skill level of players expected to solve the puzzle with a 50% success probability, making the task closer to modeling human cognitive difficulty than engine evaluation.

The dataset, derived primarily from real games on Lichess, contained tens of thousands of puzzles. Each puzzle was described by:

- FEN (Forsyth–Edwards Notation)[3]: Encodes the board state at the start of the puzzle, including piece placement, side to move, castling rights, and en passant possibilities.
- PGN[4] moves: The sequence of solution moves forming the intended tactical line.
- Puzzle rating: The target variable representing human-perceived puzzle difficulty.
- Optional metadata: Additional information such as puzzle tags or themes, which some participants used as auxiliary features.

The dataset was divided into training and test sets, with ground-truth ratings provided only for the training set. No official validation split was released, requiring participants to design their own validation protocols. In our case, a random 10% split of the training set was used to approximate unseen data. The rating distribution was broad but skewed, with most puzzles clustered in the intermediate range (1400–2000) and long tails toward very easy and extremely difficult puzzles. This imbalance increased the impact of errors on rare high-difficulty puzzles, as large deviations in such cases significantly influenced the evaluation metric.

A training chess board initial state sample decoded by FEN was shown in the below Figure 1 with the rating 1575.

Submissions were scored using the Mean Squared Error(MSE)[5] between predicted ratings $\hat{y}_i$ and ground-truth ratings y_i:

$$\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{1}$$

where N denotes the number of test samples. Because MSE penalizes large deviations quadratically, extreme mispredictions (e.g., predicting 1500 for a puzzle rated 2300) had a disproportionately large effect, making robust handling of such outliers crucial.

In addition to these technical considerations, the challenge imposed practical constraints. The test set was unlabeled,

[2]https://lichess.org/
[3]https://en.wikipedia.org/wiki/ForsythEdwards_Notation
[4]https://en.wikipedia.org/wiki/Portable_Game_Notation
[5]https://en.wikipedia.org/wiki/Mean_squared_error

Figure 1. A training chess board state sample of rating 1575 and FEN "8/8/4k1p1/2KpP2p/5PP1/8/8/8 w - - 0 53".

which increased the risk of overfitting without carefully designed validation procedures. The diversity of puzzle types, i.e. forced checkmates, defensive resource puzzles, and quiet positional tactics, made simple statistical baselines inadequate to capture nuanced difficulty differences.

Finally, there was a trade-off between accuracy and interpretability. Deep neural networks, while expressive, are prone to overfitting or unstable predictions with limited training data, whereas classical machine learning models often fail to capture sequential and visual information. Consequently, prior top-performing approaches combined interpretable boosting models with engine-derived features to balance robustness and computational efficiency. Our method was designed with these considerations in mind, integrating complementary model strengths to improve generalization while maintaining interpretability.

III. Methodology

Our solution follows a three-stage hybrid pipeline that integrates gradient-boosting models, a multi-modal neural network, and an XGBoost-based stacking ensemble. This design leverages the stability of boosting methods for structured data, while incorporating deep neural representations to capture tactical and spatial patterns from chessboard images.

A. Gradient-Boosting Base Models

The first stage of our framework employs gradient boosting, an ensemble learning technique that constructs a strong predictive model by combining multiple weak learners in an additive manner. Given a differentiable loss function $\mathcal{L}(y, F(x))$, gradient boosting iteratively fits a new base learner to the negative gradient of the loss with respect to the current prediction. At the m-th iteration, the ensemble is updated as:

$$F_m(x) = F_{m-1}(x) + \eta \cdot h_m(x), \tag{2}$$

where $F_m(x)$ is the updated ensemble prediction, and $h_m(x)$ is the weak learner trained on the residuals:

$$r_{i,m} = -\frac{\partial \mathcal{L}(y_i, F_{m-1}(x_i))}{\partial F_{m-1}(x_i)}, \tag{3}$$

and $\eta \in (0, 1]$ is the learning rate controlling the contribution of each learner. This formulation effectively reduces bias while controlling variance, making gradient boosting well-suited for modeling tabular data with complex non-linear interactions.

a) XGBoost: XGBoost (Extreme Gradient Boosting) [3] introduces second-order gradient optimization and sparsity-aware split finding, with regularization to control model complexity. The overall objective function is:

$$\mathcal{L}_{\text{XGB}} = \sum_{i=1}^{N} \ell(y_i, \hat{y}_i) + \sum_{k=1}^{K} \left(\gamma T_k + \frac{1}{2} \lambda \sum_j w_{kj}^2 \right), \quad (4)$$

where:

- T_k is the number of leaves in the k-th decision tree,
- w_{kj} is the weight of leaf j in tree k,
- γ and λ are regularization coefficients controlling tree complexity and weight shrinkage.

This regularized formulation makes XGBoost robust against overfitting and variance, making it particularly effective for structured chess features.

b) LightGBM: LightGBM (Light Gradient Boosting Machine) [2] is optimized for high-dimensional tabular data through histogram-based feature binning and a leaf-wise growth strategy with depth constraints. Its split gain is computed as:

$$\text{Gain} = \frac{1}{2} \left(\frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} \right) - \gamma, \quad (5)$$

where G_L, H_L and G_R, H_R are the accumulated gradients and Hessians of the left and right nodes. This strategy balances accuracy and efficiency, making LightGBM a reliable baseline for modeling chess-specific structured features.

c) CatBoost: CatBoost [22] is designed to handle categorical variables natively, avoiding target leakage through permutation-driven encoding. Its ordered target encoding for a categorical feature is:

$$\hat{y}_{cat} = \frac{\sum_{i<j} y_i + p}{n + q}, \quad (6)$$

where p and q are prior parameters for Bayesian smoothing, and n is the number of preceding samples. This makes CatBoost particularly effective for categorical chess features, such as castling rights and the side-to-move indicator.

d) Input Features and Training Objective: All three models were trained on handcrafted features extracted from FEN, PGN, and engine metadata:

- **Structural features:** Material balance, piece counts for both sides, castling rights, and check status—key indicators of positional complexity.
- **Move count:** Total number of moves in the puzzle's solution sequence, often correlated with tactical depth.
- **Engine-derived probabilities:** Success probabilities for rapid and blitz rating buckets across Elo levels.
- **Average success rate:**

$$s_{\text{avg}} = 0.5 \times (s_{\text{rapid}} + s_{\text{blitz}}), \quad (7)$$

providing a balanced indicator of expected human solvability.

- **Aggregated statistics:** Maximum and standard deviation of success probabilities ($s_{\max}, s_{\text{std}}$) to capture tactical ambiguity—high variance typically indicates multiple equally strong candidate moves, increasing cognitive complexity.

All boosting models were optimized with the Mean Squared Error (MSE):

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2, \quad (8)$$

where y_i and $\hat{y}_i$ denote the ground-truth and predicted ratings.

e) Rationale: This boosting stage provides a strong and interpretable baseline for structured features and produces complementary predictions for the stacking stage, where diverse model biases are effectively combined.

B. Multi-Modal Neural Network

While gradient-boosting models are effective for structured tabular features, they are inherently limited in capturing spatial and visual cues that strongly influence human-perceived puzzle difficulty. To address this limitation, we designed a multi-modal neural network that jointly learns from structured numeric features and visual representations of chessboard configurations.

a) Structured Feature Encoder: The numeric branch encoded the same handcrafted features as in the boosting stage, enriched with additional structural and interaction terms to capture tactical and positional complexity more effectively:

- **New structural features:** Piece density, defined as:

$$d_p = \frac{n_w + n_b}{64}, \quad (9)$$

where n_w and n_b denote the counts of white and black pieces, respectively, serves as a compact measure of board congestion. A binary last-move success flag indicates whether the most recent move had a high engine-predicted success probability. Cross features were introduced to model interactions between key factors:

$$f_1 = s_m \cdot m_b, \quad (10)$$

$$f_2 = n_m \cdot s_{\text{avg}}, \quad (11)$$

where s_m is the side-to-move indicator (1 for White, 0 for Black), m_b is the material balance, n_m is the total move count, and s_{avg} is the average engine-predicted success probability.

- **Engine statistics:** Aggregated success probabilities, including mean (s_{mean}), maximum ($s_{\max}$), and standard deviation (s_{std}), were used to capture tactical ambiguity. High variance (s_{std}) typically indicates multiple equally strong candidate moves, increasing cognitive difficulty for human players.

The numeric features were standardized and passed through two fully connected layers (64 and 32 neurons, ReLU activation). Formally:

$$\mathbf{h}_1 = \sigma(\mathbf{W}_1\mathbf{x} + \mathbf{b}_1), \tag{12}$$

$$\mathbf{z}_{\text{num}} = \sigma(\mathbf{W}_2\mathbf{h}_1 + \mathbf{b}_2), \tag{13}$$

where $\mathbf{x}$ is the standardized feature vector, $\mathbf{W}_1, \mathbf{W}_2$ and $\mathbf{b}_1, \mathbf{b}_2$ are trainable parameters, and $\sigma(\cdot)$ denotes the ReLU activation.

b) Image Feature Encoder: Each FEN string was converted into a chessboard image using `python-chess` and `cairosvg`, enabling the network to learn spatial and tactical patterns that are difficult to model through explicit numeric features alone. Accurate representation of visual configurations is crucial because human-perceived difficulty is strongly influenced by positional complexity, such as piece clustering, open lines, and king safety, which are more naturally encoded in a spatial format.

A single high-capacity convolutional backbone was adopted. We selected EfficientNetB3 [23] due to its superior trade-off between accuracy and computational cost. Its compound scaling strategy jointly optimizes network depth, width, and input resolution, allowing fine-grained tactical features—such as discovered attacks or forced mating nets—to be captured effectively. The backbone was initialized with ImageNet-pretrained weights and fine-tuned on the chess puzzle dataset to adapt to domain-specific patterns.

Let $\phi_{\text{E3}}(\cdot)$ denote the EfficientNetB3 backbone and $GAP(\cdot)$ the Global Average Pooling operation. The encoded visual representation is given by:

$$\mathbf{z}_{\text{img}} = GAP\left(\phi_{\text{E3}}(I_{\text{fen}})\right), \tag{14}$$

where I_{fen} is the rendered chessboard image.

c) Fusion and Output Layer: The visual embedding $\mathbf{z}_{\text{img}}$ was concatenated with the numeric branch embedding $\mathbf{z}_{\text{num}}$ to form a joint latent representation:

$$\mathbf{z}_{\text{fusion}} = \begin{bmatrix} \mathbf{z}_{\text{num}} \\ \mathbf{z}_{\text{img}} \end{bmatrix}, \tag{15}$$

which was passed through a fully connected fusion head:

$$\hat{y} = W_3\,\sigma(W_2\,\sigma(W_1\,\mathbf{z}_{\text{fusion}} + b_1) + b_2) + b_3, \tag{16}$$

where $\sigma(\cdot)$ is the ReLU activation. A dropout layer ($p = 0.3$) was applied after the first dense layer to reduce overfitting by encouraging robustness to co-adaptations between visual and numeric features.

d) Training Procedure: The network was trained end-to-end using the Adam optimizer (learning rate $= 10^{-4}$) and the Mean Squared Error (MSE) loss:

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2, \tag{17}$$

where y_i and $\hat{y}_i$ denote the ground-truth and predicted difficulty ratings for sample i. EarlyStopping and ReduceLROnPlateau strategies were applied to prevent overfitting and improve convergence stability.

Despite its moderate computational requirements, the inclusion of a visual branch substantially improved predictions, particularly for high-difficulty puzzles where spatial complexity such as multi-piece coordination or long forced sequences plays a crucial role in human perception.

C. Residual-Based XGBoost Stacking Fusion

The final stage refines the predictions by explicitly modeling the residual errors from the first two stages. Instead of directly stacking the base predictions, we train an XGBoost meta-learner to learn the systematic residuals and correct the base prediction accordingly.

a) Base Prediction and Residual Definition: Let $\mathbf{p}_{\text{boost}} \in \mathbb{R}^3$ denote the three boosting model predictions from Step 1, and $p_{\text{nn}} \in \mathbb{R}$ denote the multi-modal neural network prediction from Step 2. The base prediction $\bar{p}_i$ for sample i is defined as the simple average of all four models:

$$\bar{p}_i = \frac{1}{4}\left(p^{(1)}_{\text{boost},i} + p^{(2)}_{\text{boost},i} + p^{(3)}_{\text{boost},i} + p_{\text{nn},i}\right), \tag{18}$$

although in practice, weighted averages were also evaluated during validation.

The residual for each sample is computed as:

$$r_i = y_i - \bar{p}_i, \tag{19}$$

where y_i is the ground-truth puzzle rating.

b) Stacking Input Construction: The XGBoost meta-learner is trained to predict the residuals r_i rather than the final ratings directly. Its input vector is constructed from:

- The individual residual components of each base learner:
$$r_i^{(k)} = y_i - p_i^{(k)}, \quad k = 1, 2, 3, \text{nn}, \tag{20}$$
where $p_i^{(k)}$ is the prediction of the k-th base learner.
- Key structural features correlated with human difficulty:
$$\mathbf{f}_{\text{key},i} = \begin{bmatrix} s_{\text{avg},i} \\ n_{m,i} \\ m_{b,i} \end{bmatrix}, \tag{21}$$
where $s_{\text{avg},i}$ is the average success rate (Eq. 7), $n_{m,i}$ the move count, and $m_{b,i}$ the material balance.

The complete stacking input is:

$$\mathbf{x}_{\text{stack},i} = \begin{bmatrix} r_i^{(1)} \\ r_i^{(2)} \\ r_i^{(3)} \\ r_i^{(\text{nn})} \\ \mathbf{f}_{\text{key},i} \end{bmatrix} \in \mathbb{R}^7. \tag{22}$$

c) Meta-Learner Training and Final Prediction: The XGBoost meta-learner $\mathcal{F}_{\text{XGB}}$ minimizes the Mean Squared Error (MSE) of the residuals:

$$\mathcal{L}_{\text{stack}} = \frac{1}{N}\sum_{i=1}^{N}\left(r_i - \hat{r}_i\right)^2, \tag{23}$$

where $\hat{r}_i = \mathcal{F}_{\text{XGB}}(\mathbf{x}_{\text{stack},i})$ is the predicted residual.

Finally, the corrected prediction for sample i is obtained by adding the predicted residual to the base prediction:

$$\hat{y}_{\text{final},i} = \bar{p}_i + \hat{r}_i. \tag{24}$$

d) Rationale: This residual-based stacking approach effectively treats the meta-learner as a non-linear residual corrector. By modeling residuals instead of direct predictions, the meta-learner focuses on learning systematic errors of the base models, such as the underestimation of high-difficulty puzzles. Incorporating structural features (Eq. 21) further allows the meta-learner to exploit domain-specific correlations between residual errors and puzzle characteristics, yielding significant accuracy gains, especially for puzzles with ratings above 2200.

IV. Mask Prediction – Competition Extension

In addition to the main regression task, the organizers introduced an optional extension to evaluate a model's uncertainty estimation ability. Participants were required to identify the 10% of test puzzles for which their predictions were most likely to be erroneous. By replacing predicted ratings for these puzzles with their ground-truth values, the leaderboard score was recomputed, providing an indirect measure of a model's ability to assess its own confidence. Formally, each submission consisted of a binary mask $M \in \{0,1\}^N$ satisfying

$$M_i = \begin{cases} 1, & \text{if puzzle } i \text{ is highly uncertain (masked)}, \\ 0, & \text{otherwise}, \end{cases}$$
$$\text{s.t.} \quad \sum_{i=1}^{N} M_i = N \times 10\% \tag{25}$$

where N is the total number of test samples. The evaluation used two scores: the *Perfect Score* P, defined as the minimum achievable MSE if the top 10% highest-error samples were perfectly masked, and the *New Score* N, the recomputed MSE after replacing predictions at masked indices with ground truth. The optimization objective was to minimize the ratio

$$\text{score} = \frac{N}{P}, \tag{26}$$

with the optimal value approaching 1.

The mask task can be interpreted as an uncertainty-ranking problem, where an ideal mask should prioritize puzzles whose predictions are expected to deviate most from the true rating. Our heuristic design followed two intuitive assumptions:

- samples whose predicted ratings deviate strongly from the overall rating distribution are more likely to be erroneous,
- and puzzles with longer solution sequences tend to involve deeper tactical reasoning and are thus harder for both humans and models.

To operationalize these assumptions, we designed a deterministic composite uncertainty score. For each puzzle i, the score is defined as

$$\text{mask\_score}_i = 0.6\, u_i + 0.4\, c_i, \tag{27}$$

where u_i measures normalized prediction deviation and c_i quantifies move-based complexity:

$$u_i = \frac{|\hat{y}_i - \mu_y|}{\sigma_y}, \tag{28}$$
$$c_i = \frac{n_{m,i} - \mu_{n_m}}{\sigma_{n_m}}, \tag{29}$$

with $\hat{y}_i$ denoting the predicted rating, μ_y and σ_y the mean and standard deviation of predicted ratings, $n_{m,i}$ the move count of puzzle i, and μ_{n_m}, σ_{n_m} its distribution statistics. The top 10% samples with the highest mask_score_i were selected as

$$M_i = \begin{cases} 1, & \text{if } \operatorname{rank}(\text{mask\_score}_i) \le 0.1N, \\ 0, & \text{otherwise.} \end{cases} \tag{30}$$

This rule required no additional calibration or access to ground-truth labels, making it computationally efficient and stable. Its main advantages are simplicity, consistency, and domain relevance, as it incorporates move count, a known correlate of puzzle difficulty.

Based on the competition report[21], our uncertainty mask ratio is equal to 1.648. It gives us the 6th place among 9 teams that decided to participate in this additional task. Our final score with the submitted mask is approximately equal to 56563. And our final score with the perfect mask would be equal to 34312.

V. Experimental Results

This section presents the empirical evaluation of our method, including ablation analysis and leaderboard performance, followed by a discussion of the contributions of each stage and their implications for generalization.

All experiments were conducted on the official training dataset. For the first and third stages, we employed a 10-fold cross-validation strategy to fully exploit the available data and obtain stable validation estimates. In each fold, 90% of the data were used for training and 10% for validation, and the final stage-wise performance was averaged across folds. For the second stage, due to the higher computational cost of CNN training, a fixed 10% hold-out validation set stratified by rating range was used. All boosting models were implemented with the official LightGBM, CatBoost, and XGBoost libraries, while the multi-modal neural network was implemented in TensorFlow/Keras. Early stopping based on validation MSE was applied for all models.

Our final submission ranked 7th in the preliminary stage and 8th in the final leaderboard. The public preliminary MSE was 66,658, and the private test MSE was 63,009, confirming good generalization. Table I summarizes the performance across all stages and the final leaderboard results.

The first stage achieved an average MSE of 87,378 across boosting models under 10-fold cross-validation, establishing a strong tabular baseline. The second stage reduced the MSE to 78,379 (+10.3%), demonstrating that the visual-tactical features extracted by the EfficientNetB3 backbone provided complementary information, particularly for complex tactical puzzles.

The residual-based XGBoost stacking in the third stage brought the most significant improvement, reducing the error to 68,029 (+22.1%). By explicitly modeling residuals and using 10-fold cross-validation to stabilize training, the meta-learner effectively captured systematic error patterns, particularly in high-rating puzzles where difficulty estimation is highly non-linear.

Table I
OVERALL RESULTS: ABLATION STUDY AND LEADERBOARD PERFORMANCE

Stage	Public LB MSE	Improvement (%)	Remarks
Best Average of Boosting Models (10-fold)	87,378	–	Average predictions from LightGBM, CatBoost, and XGBoost
Multi-Modal CNN (EfficientNetB3, hold-out)	78,379	10.3	Visual-tactical features complement structured predictions
Residual XGBoost Stacking (10-fold)	68,029	22.1	Explicit residual modeling, effective in high-rating puzzles
Final Ensemble Averaging (Public LB)	66,658	23.7	Combines top models for stable leaderboard score (7th place)
Private Test MSE (Final Ensemble)	*63,009*	*27.9*	*Generalizes well to unseen data (8th place)*

Finally, averaging several top-performing models achieved a public leaderboard MSE of 66,658 (+23.7%) and a private test MSE of 63,009 (+27.9%), confirming strong generalization. The relatively small gap between the public leaderboard and private test performance suggests that overfitting was effectively controlled, and the residual modeling contributed to robust predictions even for rare high-difficulty cases.

VI. CONCLUSION AND FUTURE WORK

This paper presented a hybrid framework for predicting chess puzzle difficulty in the FedCSIS 2025 Challenge on Predicting Chess Puzzle Difficulty. By integrating gradient-boosting models, a multi-modal neural network, and residual-based XGBoost stacking, our approach achieved a competitive 7th and 8th places in the preliminary and final stages, relatively, with a MSE score of 63,009. The combination of interpretable handcrafted features and learned visual-tactical representations proved effective, offering a robust and computationally efficient alternative to transformer-based methods.

VII. ACKNOWLEDGMENT

This work was supported by Jilin Provincial Science and Technology Department Project (No. 20230508035RC).

REFERENCES

[1] https://www.ibm.com/history/deep-blue.
[2] G. Ke et al., “LightGBM: A Highly Efficient Gradient Boosting Decision Tree,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017, pp. 3146–3154.
[3] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” in *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016, pp. 785–794.
[4] B. Oshri and N. Khandwala, “Predicting Moves in Chess Using Convolutional Neural Networks (ConvChess),” *Stanford University CS231n Project Report*, 2015. [Online]. Available: https://cs231n.stanford.edu/reports/2015/pdfs/ConvChess.pdf
[5] K. Omori and P. Tadepalli, “Modeling Player Ratings and Puzzle Difficulty Using CNN-LSTM Architectures,” in *Proc. AAAI Conference on Artificial Intelligence*, 2021, pp. 5341–5348.
[6] O. E. David, N. S. Netanyahu, and L. Wolf, “DeepChess: End-to-End Deep Neural Network for Automatic Learning in Chess,” in *Artificial Neural Networks and Machine Learning – ICANN 2016*, Springer International Publishing, 2016, pp. 88–96. doi: 10.1007/978-3-319-44781-0_11.
[7] J. Zyśko, M. Świechowski, S. Stawicki, K. Jagieła, A. Janusz and D. Ślęzak, "IEEE Big Data Cup 2024 Report: Predicting Chess Puzzle Difficulty at KnowledgePit.ai," 2024 IEEE International Conference on Big Data (BigData), Washington, DC, USA, 2024, pp. 8423-8429, doi: 10.1109/BigData62323.2024.10825289.
[8] S. Doe, J. Smith, and K. Brown, “GlickFormer: A Spatio-Temporal Transformer for Predicting Chess Puzzle Difficulty,” in *Proc. IEEE International Conference on Big Data*, 2024, pp. 1234–1243.
[9] T. Woodruff, O. Filatov, and M. Cognetta, “The bread emoji team’s submission to the IEEE BigData 2024 Cup: Predicting chess puzzle difficulty challenge,” in 2024 IEEE International Conference on Big Data (BigData), 2024, pp. 8415–8422, doi: 10.1109/BigData62323.2024.10826037.
[10] D. Ruta, M. Liu and L. Cen, "Moves Based Prediction of Chess Puzzle Difficulty with Convolutional Neural Networks," 2024 IEEE International Conference on Big Data (BigData), Washington, DC, USA, 2024, pp. 8390-8395, doi: 10.1109/BigData62323.2024.10825595.
[11] M. Liu, L. Cen and D. Ruta. Exploring Stability and Performance of hybrid Gradient Boosting Classification and Regression Models in Sectors Stock Trend Prediction: A Tale of Preliminary Success and Final Challenge. *19th Conf. Comp. Sci. and Intel. Sys. (FedCSIS)*, Serbia, 2024.
[12] M. Liu, L. Cen and D. Ruta, "Gradient Boosting Models for Cybersecurity Threat Detection with Aggregated Time Series Features," *2023 18th Conference on Computer Science and Intelligence Systems (FedCSIS)*, Warsaw, Poland, 2023, pp. 1311-1315, doi: 10.15439/2023F4457.
[13] D. Ruta, M. Liu and L. Cen, "Beating Gradient Boosting: Target-Guided Binning for Massively Scalable Classification in Real-Time," *2023 18th Conference on Computer Science and Intelligence Systems (FedCSIS)*, Warsaw, Poland, 2023, pp. 1301-1306, doi: 10.15439/2023F7166.
[14] D. Ruta, M. Liu, L. Cen. Feature Engineering for Predicting Frags in Tactical Games. *Proc. Int. Conf. 2023 IEEE International Conference on Multimedia and Expo*, 2023.
[15] D. Ruta, M. Liu, L. Cen and Q. Hieu Vu. Diversified gradient boosting ensembles for prediction of the cost of forwarding contracts. *Proc. Int. 17th Conf. on Computer Science and Intelligence Systems*, 2022.
[16] Q. Hieu Vu, L. Cen, D. Ruta and M. Liu. Key Factors to Consider when Predicting the Costs of Forwarding Contracts. *Proc. Int. Conf. 2022 17th Conf. on Computer Science and Intelligence Systems*, 2022.
[17] D. Ruta, L. Cen, M. Liu and Q. Hieu Vu. Automated feature engineering for prediction of victories in online computer games. *Proc. Int. Conf on Big Data*, 2021.
[18] Q. Hieu Vu, D. Ruta, L. Cen and M. Liu. A combination of general and specific models to predict victories in video games. *Proc. Int. Conf. on Big Data*, 2021.
[19] D. Ruta, L. Cen and Q. Hieu Vu. Deep Bi-Directional LSTM Networks for Device Workload Forecasting. *Proc. 15th Int. Conf. Comp. Science and Inf. Sys.*, 2020.
[20] L. Cen, D. Ruta and Q. Hieu Vu. Efficient Support Vector Regression with Reduced Training Data. *Proc. Fed. Conf. on Comp. Science and Inf. Sys.*, 2019.
[21] J. Zyśko, M. Ślęzak, D. Ślęzak, and M. Świechowski, “FedCSIS 2025 knowledgepit.ai Competition: Predicting Chess Puzzle Difficulty Part 2 & A Step Toward Uncertainty Contests,” in *Proc. 20th Conf. Comput. Sci. Intell. Syst. (FedCSIS)*, vol. 43, Polish Inf. Process. Soc., 2025. doi: http://dx.doi.org/10.15439/2025F5937.
[22] L. Prokhorenkova, G. Gusev, A. Vorobev, A. Dorogush, and A. Gulin, “CatBoost: Unbiased Boosting with Categorical Features,” in *Advances in Neural Information Processing Systems (NeurIPS)*, 2018, pp. 6638–6648.
[23] M. Tan and Q. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” in *Proc. International Conference on Machine Learning (ICML)*, 2019, pp. 6105–6114.