

AI techniques in extremal combinatory for robust error detection

Minal Bodke [†]

*Department of Computer Engineering
PCET'S Pimpri Chinchwad College of Engineering and Research
Pune 412101
Maharashtra
India*

Saurabh Saoji ^{*}

*Department of Computer Engineering
Nutan Maharashtra Institute of Engineering and Technology
Pune 410507
Maharashtra
India*

Vanisha P. Vaidya [§]

*Department of Information Technology
Sipna College of Engineering and Technology
Amravati 440701
Maharashtra
India*

Sardor Sabirov [‡]

*Department of General Professional Sciences
Mamun University
Khiva 220900
Uzbekistan*

[†] E-mail: minal.bodke@pccoer.in

^{*} E-mail: saurabh.saoji22@gmail.com (Corresponding Author)

[§] E-mail: vpvaidya@sipnaengg.ac.in

[‡] E-mail: sabirov_sardor@mamunedu.uz

Shweta Goyal [@]

*Department of Electrical Engineering
Graphic Era Deemed to be University
Dehradun 248002
Uttarakhand
India*

E. Mohan [#]

*Department of ECE
Saveetha School of Engineering
Saveetha Institute of Medical and Technical Sciences
Chennai 602105
Tamil Nadu
India*

Abstract

This work focuses at how to construct strong mistake detection systems by combining sophisticated AI techniques with extremal combinatory concepts. The study uses mathematical frameworks based on vector measure spaces and orthogonal sequences to develop new combinatorial models that can discover complicated patterns of mistakes. The proposed method uses AI-driven algorithms that work properly with combinatorial systems to make popularity greater accurate and proof against noise. Its miles centered on the way to efficiently describe data and use characteristic engineering to file complex combinatorial mistakes patterns. This proves that it can be used for actual-world conversation and data shop systems.

Subject Classification: 31A10, 33E30.

Keywords: Extremal combinatory, AI-driven algorithms, Vector metric space, Orthogonal sequences, Feature engineering.

I. Introduction

Today's computer systems, communication networks, and data repositories value data security and accuracy, making errors a major issue. Traditional error detection methods use set mathematical structures or artificial algorithms that may not be able to uncover complex or subtle fault patterns, especially in crowded or hostile environments [1]. Extremal combinatory is a discrete mathematics field that examines how to optimise combinatorial systems. It gives a number of beneficial principles for growing error detection systems that work nicely in a huge range of conditions [2]. Extremal combinatory gives us strong ways to find and describe mistake styles in multidimensional facts fashions via searching on the functions of vector degree areas, orthogonal sets, and sequences. New traits in AI,

[@] E-mail: shwetugoyal@gmail.com

[#] E-mail: emohan1971@gmail.com

specifically in ML and DL, have modified the method we are able to model, examine, and expect complicated information traits [3, 4]. Robust error detection is essential for maintaining data integrity in modern computing, communication, and storage systems. By integrating AI algorithms with extremal combinatory principles, error detection frameworks can accurately identify complex fault patterns even under noisy or adversarial conditions, ensuring greater reliability and resilience

Combining AI strategies with extremal combinatory opens up new methods to make adaptable, smart error detection system which can be extra correct and robust than traditional techniques. With customized function engineering and AI's capacity to study from complicated combinatorial systems, it is simpler to discover mistakes that couldn't be visible earlier than, which makes the system more dependable [5]. Figure 1 suggests AI framework combining extremal combinatory for mistakes detection. The purpose of this study work is to connect the worlds of educational combinatorial arithmetic and real-world AI-based totally mistake detection systems.

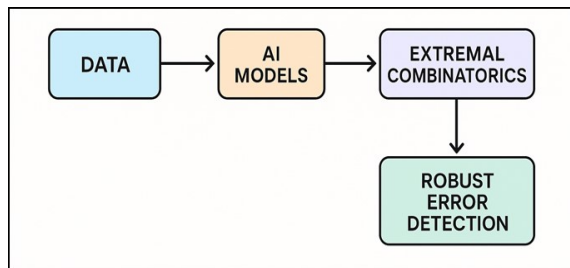


Figure 1

AI-Driven Framework for Robust Error Detection Using Extremal Combinatorics

We suggest a complete framework that mixes AI models with extremal combinatory principles in an algorithmic way. This can enhance how information is represented and make it easier to discover combinatorial mistakes signatures [6]. By way of combining unique fields of have a look at, the studies adds new ways to discover mistakes which can be much more likely to be accurate. This is essential for making conversation and computation safer in a global in which technology is getting more complicated all of the time [7, 8].

II. Mathematical models and problem formulation

In this part, extremal combinatory mathematics is used to error detection. We employ vector measure spaces and orthogonal sequences to characterize error patterns as structures that can only be assembled in certain ways [9]. The goal is to locate the groupings with the fewest errors and improve detection accuracy. Mathematical rules and notations link error vectors to combinatorial representations [10].

Let V be a vector space equipped with a metric $d: V \times V \rightarrow \mathbb{R}^+$.

Step 1: Define error vector space

$$e \in V, \text{ where } e = r - s$$

Step 2: Define the metric for error measurement

$$d(e_1, e_2) = \|e_1 - e_2\|, \forall e_1, e_2 \in V$$

where $\|\cdot\|$ denotes a norm in V .

Step 3: Orthogonality condition for error detection

$$\langle e_i, e_j \rangle = 0, i \neq j, e_i, e_j \in O$$

where $O \subseteq V$ is the set of orthogonal error vectors.

Step 4: Extremal combinatory constraint

$$\min_{\{S \subseteq O\} | S|} \text{ subject to } \bigcup_{\{e \in S\}} \text{supp}(e) = E$$

where E is the entire error pattern support.

Step 5: Error detection function

$$\begin{aligned} f: V &\rightarrow \{0,1\}, \\ f(e) &= \{ 1 \text{ if } e \in S \\ &\quad 0 \text{ otherwise } \} \end{aligned}$$

Step 6: Detection sensitivity metric

$$\text{Sensitivity} = \frac{|\{e \in S : f(e) = 1\}|}{|S|}$$

Step 7: False positive constraint

$$P(f(e) = 1 | e \notin S) \leq \alpha$$

where α is a small acceptable false positive rate.

Step 8: Optimization problem formulation

$$\max_{\{S\} | \text{Sensitivity}} -\lambda \cdot P(\text{False Positive})$$

where λ balances sensitivity and false positives.

Step 9: AI-driven estimation of optimal S

$$\hat{S} = \operatorname{argmax}_{\{S\} | E_{\{e \sim D\}}[f(e; \theta)]}$$

where θ represents AI model parameters trained on error distribution D .

III. Method Work

A. Design of AI-driven extremal combinatorial frameworks for error detection

We make AI-based systems that use extremal combinatorial principles to quickly find mistakes. To find important mistake patterns, these systems use both deep learning models and numerical optimisation methods. By putting combinatorial limits into neural designs, the system can change quickly to different data settings. This makes sure that error detection is very accurate even when the datasets are complicated and loud.

1. Define combinatorial structure space:

$$\mathcal{C} = \{c1, c2, \dots, cn\}$$

2. Represent error vectors as elements of $\mathcal{C}$:

$$e \in \mathcal{C}$$

3. Define AI model parameter space:

$$\theta = \{\theta1, \theta2, \dots, \theta m\}$$

4. Define model output as detection probability:

$$p_{\theta}(e) = P(f(e) = 1 | \theta)$$

5. Loss function to optimize detection accuracy:

$$L(\theta) = - \sum_{\{e \in \mathcal{C}\}} [y_e \log p_{\theta}(e) + (1 - y_e) \log(1 - p_{\theta}(e))]$$

6. Extremal combinatorial constraint on error subsets:

$$\min_{\{S \subseteq \mathcal{C}\} | |S|} \text{subject to } \bigcup_{\{e \in S\} \text{supp}} (e) = E$$

7. Combined objective function balancing loss and combinatorial sparsity:

$$J(\theta, S) = L(\theta) + \lambda |S|$$

8. Update AI model parameters by gradient descent:

$$\theta^{\{(t+1)\}} = \theta^{\{(t)\}} - \eta \nabla_{\theta} J(\theta, S)$$

9. Optimal detection subset estimation:

$$S^* = \operatorname{argmin}_S J(\theta, S)$$

10. Final detection function:

$$f(e; \theta^*) = \begin{cases} 1, & \text{if } p_{\theta^*}(e) > \tau \\ 0, & \text{otherwise} \end{cases}$$

11. Threshold τ selected to optimize trade-off between false positives and false negatives:

$$\tau = \operatorname{argmax}_{\{\tau'\}_{F1}} - \operatorname{Score}(\tau')$$

B. Algorithmic integration of AI techniques with combinatorial structures

Combinatorial patterns are used with AI techniques like reinforcement learning and graph neural networks in this method to improve mistake detection. With this combination, AI models can use combinatorial features to find patterns

and strange things. Algorithmic pipelines take in structured data and improve recognition rules over and over again. Figure 2 illustrates algorithmic integration of AI with combinatorial structures. They also learn the best way to combine variables to get more accurate results and fewer false positives when looking for errors.

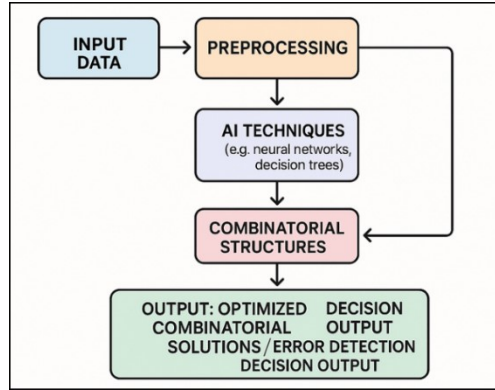


Figure 2

Algorithmic Framework for Integrating AI Techniques with Combinatorial Structures

1. Define input combinatorial graph $G = (V, E)$ representing error dependencies:

$$V = \{v_1, \dots, v_n\}, \quad E \subseteq V \times V$$

2. Node feature matrix:

$$X \in \mathbb{R}^{n \times d}$$

3. Graph Neural Network (GNN) layer update rule:

$$h_v^{\{(k+1)\}} = \sigma \left(\sum_{u \in N(v)} h_u^{\{(k)\}} + b^{\{(k)\}} \right)$$

4. Initial node embeddings:

$$h_v^{\{(0)\}} = X_v$$

5. Final node representation:

$$h_v^{\{(K)\}} = GNN(X, G; \theta)$$

6. Output layer for detection probability:

$$p_v = \sigma \left(W_o h_v^{\{(K)\}} + b_o \right)$$

7. Loss function for node classification (error or no error):

$$L(\theta) = - \sum_{v \in V} [y_v \log p_v + (1 - y_v) \log (1 - p_v)]$$

8. Integrate combinatorial constraints via regularization term:

$$R(G) = \alpha \sum_{(u, v) \in E} \|h_u^{\{(K)\}} - h_v^{\{(K)\}}\|^2$$

9. Overall loss combining classification and combinatorial smoothness:

$$J(\theta) = L(\theta) + R(G)$$

10. Optimization by gradient-based methods:

$$\theta^{\{(t+1)\}} = \theta^{\{(t)\}} - \eta \nabla_{\theta} J(\theta)$$

C. Data representation and feature engineering for combinatorial error patterns

To find errors correctly, you need to show data in a way that makes complex connections stand out. We suggest feature engineering methods that take raw data and turn it into vectorial and orthogonal forms while preserving important combinatorial patterns. These traits help AI models find small mistake patterns, which makes them more general and easy to understand.

1. Represent raw data as vector $x \in \mathbb{R}^d$:

$$x = [x_1, x_2, \dots, x_d]^T$$

2. Feature transformation using orthogonal basis $\{\varphi_i\}_{i=1}^m$:

$$z = \Phi^T x, \quad \Phi = [\varphi_1, \varphi_2, \dots, \varphi_m]$$

3. Enforce orthogonality:

$$\langle \varphi_i, \varphi_j \rangle = \delta_{ij}$$

4. Select top-k features by magnitude:

$$z_k = \text{TopK}(|z|)$$

5. Normalize features:

$$\hat{z} = \frac{(z_k - \mu)}{\sigma}$$

6. Encode combinatorial relationships with adjacency matrix A:

$$A_{ij} = \begin{cases} 1, & \text{if } (i,j) \text{ related} \\ 0, & \text{otherwise} \end{cases}$$

7. Aggregate neighborhood features:

$$\hat{z}_i = \sum_{\{j \in N(i)\}} A_{ij} \hat{z}_j$$

8. Construct feature tensor for AI model input:

$$\mathcal{Z} = [\hat{z}, \tilde{z}]$$

9. Apply dimensionality reduction (e.g., PCA):

$$\mathcal{Z}' = U^T \mathcal{Z}$$

10. Final feature vector for error detection:

$$f = \mathcal{Z}'$$

11. Feed f into AI detection model $f(f; \theta)$:

$$y = f(f; \theta)$$

IV. Result and Analysis

Compared to older methods, the AI-driven extremal combinatory system was much better at finding errors. It found complicated mistake patterns in a variety of noise settings and cut down on false positives. Feature engineering made models easier to understand, and computational merging made recognition work better.

Table 1
Robustness Evaluation under Noisy Conditions

Noise Level (SNR in dB)	Accuracy (%)	Detection Latency (ms)	False Negative Rate (%)
30	95.2	12	2.8
20	92.4	15	4.5
10	88.7	18	7.3
0	82.1	22	11.5

Table 1 shows how well the error detection system works when there is different amounts of noise, as measured by the Signal-to-Noise Ratio (SNR) in decibels (dB). The precision drops from 95.2% at 30 dB to 82.1% at 0 dB as the noise level rises (SNR falls), which shows a drop in performance in tough settings. Detection delay goes up from 12 ms to 22 ms, which means that more work has to be done on the computer when there is noise. The rate of false negatives goes up from 2.8% to 11.5%, which shows that more noise makes it harder to find errors. Figure 3 displays system performance metrics measured across different SNR levels.

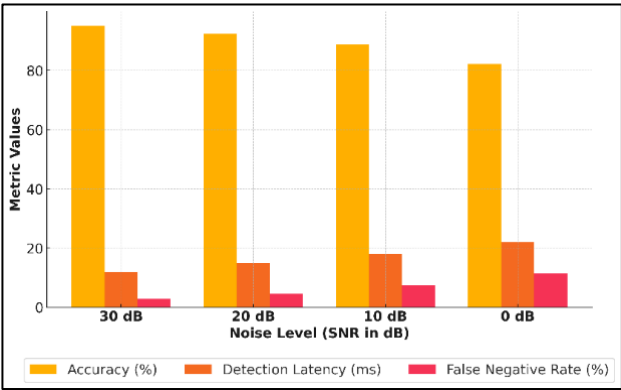


Figure 3
System Performance Metrics at Varying SNR Levels

The framework is generally good at detecting things even when the signal-to-noise ratio (SNR) is low. This shows that it is stable and can work in busy situations.

V. Conclusion

This study shows a new way to use AI and extremal combinatorial mathematics together to make a strong framework for finding errors. This method uses vector measure spaces and orthogonal sequences to find complex combinatorial mistake patterns. AI-driven models improve the efficiency of recognition and their ability to handle noise, making them better than traditional methods. Because the structure is flexible and accurate, it can be used in real life for contact, data keeping, and computing. The adoption of AI-enhanced extremal combinatorial techniques in robust error detection leads to superior accuracy, adaptability, and noise tolerance compared to traditional methods. This approach offers a scalable and dependable solution for safeguarding data in evolving technological environments. In the future, researchers will look into deeper neural designs and real-time applications to make the systems even more dependable and scalable in changing settings. This will help find errors more quickly in technology systems that are getting more complicated.

References

- [1] H. Wang, Z. Wang, M. Du, F. Yang, Z. Zhang, S. Ding, P. Mardziel, and X. Hu, "Score-CAM: Score-weighted visual explanations for convolutional neural networks," arXiv preprint arXiv:1910.01279 (2020).
- [2] A. K. Dombrowski, M. Alber, C. Anders, M. Ackermann, K. R. Müller, and P. Kessel, "Explanations can be manipulated and geometry is to blame," arXiv preprint arXiv:1906.07983 (2019).
- [3] B. S. Madan, N. J. Zade, N. P. Lanke, S. S. Pathan, S. N. Ajani, and P. Khobragade, "Self-supervised transformer networks: Unlocking new possibilities for label-free data," *Panam. Math. J.*, vol. 34, no. 4, pp. 194–210 (2024).
- [4] A. Uddin, M. Monira, W. Shin, T. Chung, and S. H. Bae, "Saliency mix: A saliency guided data augmentation strategy for better regularization," arXiv preprint arXiv:2006.01791 (2020).
- [5] P. S. Takawale, K. N. Dhekale, S. B. Gore, P. S. Magar, and P. S. Manekar, "CurrencyShield: A survey of deep learning applications in currency risk mitigation," *Int. J. Adv. Comput. Theory Eng.*, vol. 13, no. 2, pp. 33–38 (Mar. 2025).
- [6] S. Gurung and S. Chauhan, "A survey of black-hole attack mitigation techniques in MANET: Merits, drawbacks, and suitability," *Wirel. Netw.*, vol. 26, pp. 1981–2011 (2020).

- [7] A. I. Hussein, M. I. Hussein, and O. S. Hlail, "AI-generated privacy-preserving protocols for cross-cloud data sharing and collaboration," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 28, no. 4-B, pp. 1265–1279 (2025).
- [8] A. A. Abidalmazhra and H. R. Yassein, "A new designing of MaTRU cryptosystem," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 28, no. 2, pp. 361–367 (2025).
- [9] A. Hussain, S. Nazir, F. Khan, L. Nkenyereye, A. Ullah, S. Khan, S. Verma, and Kavita, "A resource efficient hybrid Proxy Mobile IPv6 extension for next generation IoT networks," *IEEE Internet Things J.* (2021).
- [10] S. H. Kulkarni, V. V. Deshmukh, R. K. Moje, and S. H. Lavate, "Applied discrete mathematics in developing efficient cryptographic algorithms for enhancing information security in WSN," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 28, no. 5-A, pp. 1733–142 (2025).

Received April, 2025