Billiards Analyzer: A Hybrid System Combining Computer Vision and Large Language Models for Strategic Shot Recommendation

Guiyin Tian
The University of Hong Kong
guiyin_tian@connect.hku.hk

Junhao He
The University of Hong Kong
hjh2003@connect.hku.hk

Abstract

This paper introduces a hybrid system for real-time strategic analysis in American Nine Ball, integrating the YOLOv5 object detector and a large language model (LLM) through prompt engineering. By leveraging top-view images of pool tables, the system detects balls and pockets, computes spatial relationships (e.g., angles, cushion proximity), and generates actionable shot recommendations through an LLM. Experimental results demonstrate that the proposed framework achieves an mAP@0.5 of 0.963 for object detection and provides context-aware strategic advice with constrained LLM randomness. This work addresses the lack of affordable, automated coaching tools in cue sports and highlights the effectiveness of combining traditional CV with LLMs for domain-specific decision-making.

1. Introduction

Billiards, particularly American Nine Ball, demands strategic decision-making based on real-time spatial analysis of ball positions and pocket layouts. While technologies like unmanned billiard halls and automatic scoring systems have emerged, affordable, AI-driven coaching tools remain scarce, especially for niche variants like Nine Ball. Existing solutions either rely on manual input or fail to integrate computer vision (CV) with domain-specific reasoning.

1.1. Two-stage framework

This study presents a two-stage framework:

1.1.1 **CV Stage:** A YOLOv5 model is trained to detect balls (ball number 1–9) and pockets in top-view images, with dataset fusion and loss function adjustments to enhance recall.

1.1.2 **LLM Stage:** A rule-constrained LLM generates shot recommendations by interpreting CV outputs (e.g., normalized coordinates, angles) alongside pre-defined Nine Ball rules (e.g., "always strike the lowest-numbered ball first").

Unlike general multimodal models (e.g., CLIP), which struggle with niche object detection and require extensive fine-tuning, our hybrid approach leverages the efficiency of CV for fixed-scene detection and the reasoning power of LLMs for strategic analysis. This avoids the limitations of vague image descriptions from multimodal models and enables precise, rule-compliant outputs.

2. Workflow

The system workflow is illustrated in Figure 1. Users upload a top-view image of the pool table, which is processed by YOLOv5 to generate bounding boxes for balls and pockets. A Python script then computes geometric features, including:

- Relative coordinates of objects in normalized image space
- Angles between the cue ball, target ball, and pocket (Eq. 1)
- Binary flags for balls within 5 cm of table cushions

These features are encoded into a JSON schema and concatenated with a system prompt defining Nine Ball rules, forming the input for the LLM API. The LLM generates a structured report prioritizing legal shots, alternative strategies, and defensive tips.

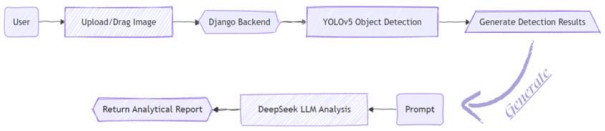

Figure 1: The workflow of the system

3. Methodology

3.1. Dataset Construction

To ensure the accuracy and robustness of the YOLOv5s model for billiard table detection, we determined that a comprehensive dataset containing thousands of annotated images would be required. During our investigation of publicly available datasets on Roboflow, we identified

separate collections containing either billiard ball annotations (Figure 3) or pocket annotations (Figure 2), but no unified dataset containing both classes. This presented a significant challenge as our computer vision system requires simultaneous detection of both balls and pockets for effective gameplay analysis. While initial consideration was given to simple dataset concatenation, this approach was rejected due to potential negative sampling effects. In such a scenario, unlabeled objects present in images (e.g., pockets in ball-focused datasets or balls in pocket-focused datasets) could be misinterpreted as background features by the model during training, potentially degrading detection performance through conflicting feature learning.

To address this challenge, we developed a novel dataset augmentation strategy:

1. Initially trained a specialized YOLOv5s model exclusively on the pocket-annotated dataset.
2. Employed this model to detect pockets in images from the ball-focused dataset
3. Converted the model's output (bounding box coordinates in [buttom left coordinate, top right coordinate, width, height] format with confidence scores) to standard YOLO annotation format
4. Merged these generated pocket annotations with existing ball annotations through rigorous coordinate validation

This methodology ensures that all training images contain complete annotations for both object classes while maintaining spatial consistency. The resulting composite dataset (demonstrated in Figure 4) preserves the original ball annotations while incorporating machine-validated pocket positions, effectively eliminating negative sampling conflicts.

Figure 2

Figure 3

Figure 4

The combined dataset has 16 color balls where American Nine Ball only consised of 9 color balls, to prevent misdetection of ball that is not in American Nine Ball, the included class in dataloader is modified to make sure the training only contain the balls that are existed in American Nine Ball.

3.2. Model Training

Due to the limited scene diversity in the dataset, the YOLOv5s model demonstrated near-perfect detection accuracy during initial training. After 100 training epochs, the model achieved a mean Average Precision (mAP@0.5) of 0.958 (Figure 5), indicating strong localization and classification performance. However, the recall rate of 0.91 revealed persistent missed detections, particularly for billiard balls and table pockets.

To address this limitation, we implemented a targeted loss function (Eq. 1) modification by introducing augmented penalties for object detection failures [1]. Specifically, the object loss component λ_2, was weighted to impose stronger gradients when the model failed to detect key elements (balls/pockets), the penalty value will be increased.

$$Loss = \lambda_1 \cdot L_{cls} + \lambda_2 \cdot L_{obj} + \lambda_3 \cdot L_{loc} \quad (1)$$

This adjustment yielded measurable improvements: the recall rate increased by 1.0 percentage points (0.91 → 0.92), corresponding to a reduction in missed detections, while maintaining high precision as evidenced by the

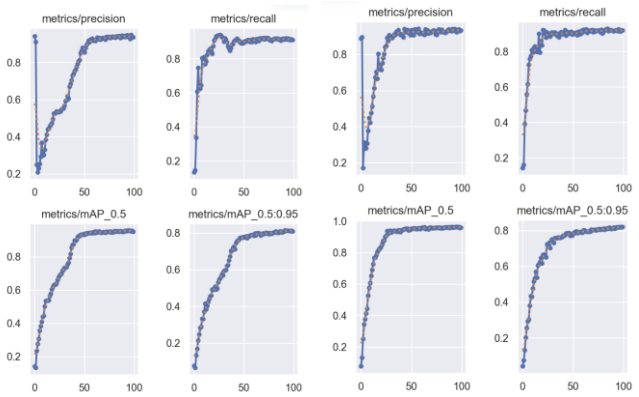

Figure 5

Figure 6

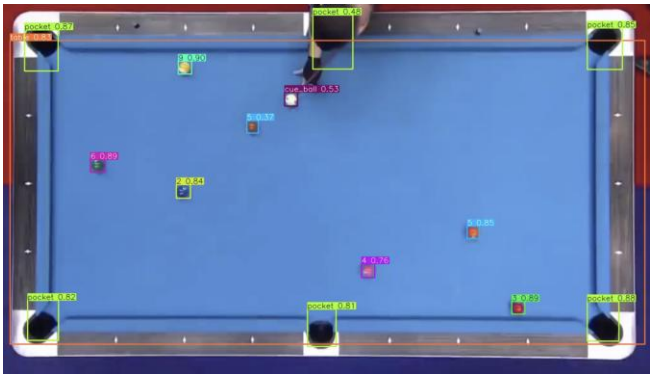

Figure 7

mAP@0.5 improvement to 0.963 (Figure 6). Figure 7 provides comparative visualization of detection outputs, demonstrating enhanced model sensitivity to small targets after optimization.

3.3. Prompt Engineering

A prompt is an input to a Generative AI model, that is used to guide its output [2]. The Yolov5 detection outputs will transform into three key features:

3.3.1 Geometric Formalization

Raw detection outputs from YOLOv5s are processed through a geometric formalization pipeline

- *Relative coordinate normalization*: Absolute pixel coordinates are converted to table-relative positions $(x', y') \in [0,1]^2$ using holography calibration
- *Angular relationship modeling*: Attack angles $\theta \in (-180°, 180°]$ between cue ball, target ball, and pocket are computed through vector decomposition
- *Cushion proximity analysis*: Ball-edge interactions are quantified using signed distance fields (SDF) with 5 cm detection threshold

3.3.2 Tactical Contextualization

Complementing the vision data, the system encodes domain-specific knowledge following World Pool-Billiard Association (WPA) standards:

- Turn sequence validation (lowest-numbered ball priority)
- Legal shot verification (cue ball-first contact checks)
- Positional difficulty coefficients (empirically calibrated through professional gameplay analysis)
- Attack/defense transition thresholds based on angular configurations

3.3.3 Prompts

Figure 8. Shows the part of JSON-formatted prompts


```
"table_state": {
  "dimensions": {
    "left": 72.5,
    "right": 1398.0,
    "top": 84.0,
    "bottom": 739.0
  },
  "pockets": [
    {
      "id": 1,
      "type": "Top-right pocket",
      "position": [
        1398.0,
        101.0
      ],
      "diameter": 80.0
    },
```

```
"balls_analysis": [
  {
    "id": "9",
    "type": "object",
    "position": [
      585.5,
      194.5
    ],
    "cue_relationship": {
      "distance": 318.0,
      "angle_to_pockets": [
        {
          "pocket_id": 1,
          "attack_angle": 137.4,
          "path_analysis": {
            "cue_to_target": {
              "distance": 318.0,
              "blockers": [
                "3"
              ]
            },
            "target_to_pocket": {
              "distance": 817.9,
              "blockers": []
            }
          },
          "clear_shot": false
```


Figure 8

3.4. Web Design

3.4.1 Overall Architecture

The web application is built on the Django framework, which follows the Model - View - Controller (MVC) architectural pattern, known as Model - View - Template (MVT) in Django. The Django backend is responsible for handling user requests, image uploads, model invocations (YOLOv5 and LLM), and data processing. The frontend uses HTML, CSS, and JavaScript to create an interactive and user - friendly interface.

3.4.2 User Interface (UI) Design

Home Page (Figure 9): The home page features a prominent drag - and - drop area for users to upload pool table images. A brief introduction to the system's functionality and a "How it Works" section are also provided to guide new users. The page layout is clean and minimalist, with a modern color scheme to enhance readability and visual appeal.

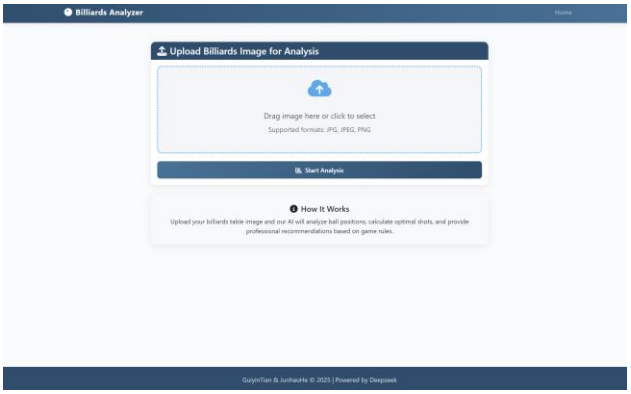

Figure 9

Analysis (Result) Page (Figure 10): After uploading an image, the user is redirected to the analysis page. This page displays the uploaded image with bounding boxes drawn around detected balls and pockets by the YOLOv5 model. Below the image, the LLM - generated analysis report is presented in a structured format, including shot recommendations, alternative strategies, and foul prevention tips. Each recommendation is accompanied by a brief explanation to help users understand the reasoning behind it. If the upload picture does not contain any Billiards information, as shown in Figure 11, the page will show the error information to remind user to reupload the picture which contains the Billiards.

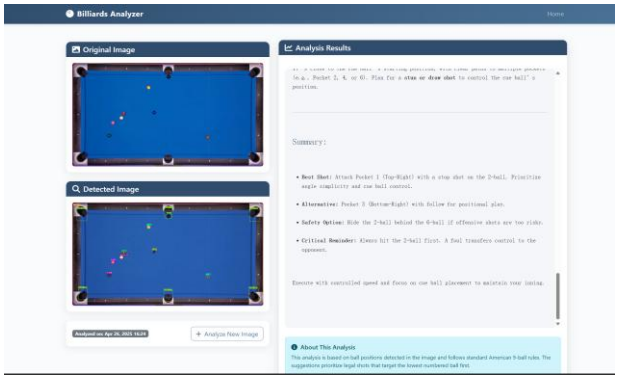

Figure 10

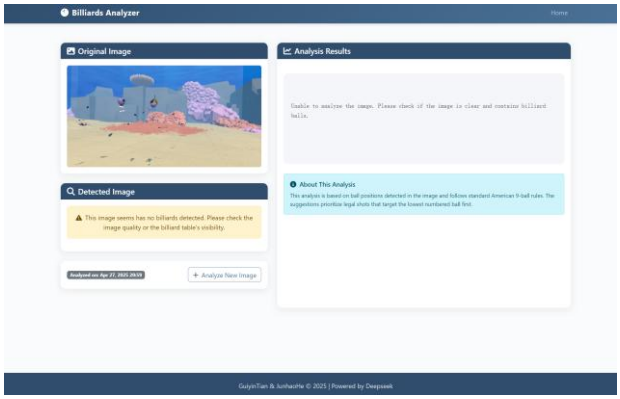

Figure 11

4. Result

4.1. Yolov5 Performance

Table 1 summarizes detection metrics on the test dataset:

Model	Precision	Recall	mAP@0.5
Baseline	0.928	0.911	0.958
Use DCNv3 in backbone	0.920	0.904	0.953
Modify loss function	0.924	0.925	0.963

Table 1

4.2. Limitations

4.2.1 Perspective Dependency in Visual Input

The current system inherits a critical constraint from its training data: exclusive reliance on top-view imagery of billiard tables. Non-orthographic perspectives induce projective distortions that degrade YOLOv5s detection accuracy, subsequently propagating errors to downstream LLM analysis. Real-world deployment scenarios requiring dynamic camera angles remain unsupported without retraining on multi-perspective annotated data.

4.2.2 Randomness of LLM Output

The output from the language model can vary even when the input prompt remains the same. To address this issue, we incorporate the rules of American Nine Ball and impose specific output restrictions at the beginning of the input prompt. While this method helps to constrain the randomness of the output to some extent, it does not completely resolve the problem.

4.2.3 API Processing Time

The large language model is accessed via an API, resulting in lengthy response times. One potential solution is to deploy the model locally, which could significantly speed up response times. However, the size of the model is limited by the performance capabilities of the local device.

4.3. Conclusion

This work presents a way to combine computer vision model and large language by using prompt engineering. This method only requires image dataset which is more affordable. The performance can be further improved by fine-tuning the large language model using a more textual dataset about the tricks of billiards, which enables the large language model to provide more useful information of how to play in the analytical report. Finally, the source code of the whole project is made available at GitHub: *https://github.com/GuiyinTIAN/Billiards-Analyzer.* If you have difficulty running the projects, feel free to email us.

Acknowledgements

We are grateful to Professor Evan Peng for his engaging lectures and support throughout the course. We would also like to acknowledge the assistance of all the Teaching Assistants from ELEC4544 for their invaluable help and guidance.

Contribution

He Junhao: Dataset Processing, Yolov5 Model Training and Optimizing, Prompt Engineering.

Tian Guiyin: Prompt Engineering, LLM API Processing, Website Designing

Each person has the same contribution to the project.

References

[1] Khanam, Rahima and Muhammad Hussain. “What is YOLOv5: A deep look into the internal features of the popular object detector.” *ArXiv* abs/2407.20892 (2024): n. pag.

[2] Schulhoff, Sander et al. “The Prompt Report: A Systematic Survey of Prompt Engineering Techniques.” (2024).

[3] Wang, Wenhai et al. “InternImage: Exploring Large-Scale Vision Foundation Models with Deformable Convolutions.” *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022): 14408-14419.