Predicting the Next Basketball Action with Graph Attention Networks and Large Language Models

Kyle Ng
Tufts University

Abstract

This paper evaluates whether a Graph Attention Network (GAT) can predict the next basketball action from NBA play-by-play data given the current game context and recent possession events. Each event is represented as a fully connected interaction graph over the ten on-court players, where node feature vectors combine learned player identity embeddings with engineered contextual and temporal features (e.g., score margin, clock, possession progress, and short event history). The resulting graph representation is used to predict the next game-level event label from a fixed action set. As an unstructured baseline, we evaluate a Large Language Model (LLM) (Mistral 7B) using zero-shot and few-shot prompting over textual play-by-play descriptions. Models are evaluated using Top-1 accuracy, Top-3 accuracy, macro-F1, and per-class confusion matrices. Results suggest that the LLM baseline captures coarse game flow but struggles with fine-grained event prediction, while the GAT learns useful relational structure and achieves stronger overall performance.

1 Introduction

Basketball is a highly interactive, multi-agent system in which the outcome of each possession is determined from coordinated decisions among players, coaches, and officials. Traditional basketball analytics often reduce this complexity to box-score summaries such as shooting percentages or aggregate counts, making it difficult to reason about how micro-context—who is on the court, recent actions, score margin, and game clock—shapes what happens next. With the increasing availability of detailed play-by-play and tracking data, recent advances in deep learning have enabled basketball to be modeled as a structured prediction problem over interacting agents rather than as a sequence of independent events [Zhang, 2020; Wu et al., 2021].

Recent research has leveraged this structure using graph-based and attention-driven models. Prior work has shown that representing players as nodes and their interactions as edges allows graph neural networks (GNNs) to capture coordinated behavior, game tempo, and tactical intent [Zhang, 2020; Li & Jiang, 2025]. Graph attention mechanisms have been effective in modeling dynamic player relationships and temporal dependencies, outperforming sequence-only baselines on tasks such as tactical recognition, player performance forecasting, and game outcome prediction [Luo & Krishnamurthy, 2023; Zhao et al., 2025; Li & Jiang, 2025; Wu et al., 2021]. Collectively, these studies suggest that explicit relational inductive biases are well suited for basketball's multi-agent dynamics. However, much of the existing work focuses on long-horizon targets—such as per-game statistics, tactical classes, or win-loss outcomes—collapsing many fine-grained decisions into a single label.

In this project, we instead focus on short-horizon next-event prediction, where the goal is to predict what happens immediately next in an NBA possession given the current game context and recent play-by-play history. Rather than predicting a specific player's next action, we model the next game-level event, which includes on-court actions (shots, rebounds, fouls) as well as administrative or strategic events (timeouts, substitutions, ejections). This formulation reflects the structure of play-by-play logs, where many meaningful events are not attributable to a single player's intention but still determine possession flow.

To model this problem, each event is represented as an interaction graph over the ten on-court players, where nodes correspond to players and edges represent potential interactions between all player pairs. We adopt this player-only graph design because players are the central decision-making agents consistently observable in play-by-play data, while other potential entities (e.g., the ball or spatial tracking states) are not reliably available in the dataset. Node features combine learned player identity embeddings with engineered contextual and temporal features that encode lineup configuration, possession progress, and short event history. A Graph Attention Network (GAT) processes these graphs to predict the next event label.

To contextualize the benefits of explicit relational structure, we additionally evaluate a Large Language Model (LLM) baseline that performs zero-shot and few-shot next-event prediction from textual play-by-play descriptions. Across multiple evaluation metrics, the graph-based model provides more reliable fine-grained next-action predictions than the unstructured language baseline, particularly for events governed by possession structure and game context.

2 Methods

We model next-action prediction in basketball as a structured multi-agent learning problem. Each play-by-play event is represented as a graph snapshot over the ten on-court players, where node features encode player identity, event participation, and contextual game state (e.g., time remaining, score margin, possession progress, and recent event history). A Graph Attention Network (GAT) is trained to predict the next event label in the possession sequence. In addition, we evaluate an LLM baseline that predicts the next action directly from textual play-by-play descriptions, enabling comparison between explicit relational modeling and unstructured language-based prediction.

Workflow Overview:

1.) Segment game into possessions
2.) Reconstruct 10-player lineup per event
3.) Build fully-connected player graph + node features
4.) Train GAT to predict next event label
5.) Evaluate on held-out possessions
6.) Compare to LLM prompting baseline

2.1 Task Definition

Given a sequence of play-by-play events within a possession, the task is to predict the next labeled event from a fixed set of action classes. Each event is associated with game context (e.g., time remaining, score margin), player participation, and on-court lineup configuration. The model observes information available up to time $t$and predicts the next event label at time $t+1$. All events are mapped into a standardized action label space (Section 2.3) to support multi-class classification

2.2 Dataset, Preprocessing and Feature Engineering

We use NBA play-by-play logs from the 2000–2001 season, obtained from Sports-Statistics.com [Sports-Statistics.com, n.d.]. The dataset contains event-level records including timestamps, event types, team identifiers, and player participation fields, which enable reconstruction of possession sequences and on-court lineups. From these logs, we construct possession-level sequences and define next-event prediction as a 12-class classification problem.

Because play-by-play logs do not provide full tracking trajectories, we focus on information consistently available in event metadata and lineup structure. To represent the game state at each event, we construct graph snapshots with node features that combine (1) learned embeddings for high-cardinality identities (e.g., Player ID) and (2) engineered contextual features derived from play-by-play context (e.g., time remaining, score margin, possession progress). This hybrid representation is motivated by prior work using GNN and attention-based methods for sports prediction, where learned embeddings capture entity-level tendencies while structured features encode context and temporal state [Zhang, 2020; Luo & Krishnamurthy, 2023; Zhao et al., 2025].

(i)Possession Segmentation: Each game is segmented into possessions using event-type heuristics and period boundaries. Much as how texts are the building blocks for sentences, it makes sense to divide games into possessions, so the model can learn structure and what happens in a real game.

(ii) Lineup Reconstruction: For each event, the full on-court lineup was attempted to be reconstructed, yielding a fixed set of ten player nodes per graph snapshot. When subs are called in, they are replaced in the lineup. If possessions had

incomplete or ambiguous lineups, they were excluded from graph construction.

(iii) Player Embeddings: Each player is represented by a learned embedding that captures individual tendencies across games and contextual situations.

(iv) Role Indicators: Role features are included to identify the primary, secondary, and tertiary participants involved in each event.

(v) Offense–Defense Encoding: Each node includes an offense–defense indicator derived from possession context, enabling differentiation of player roles conditioned on team control.

(vi) Contextual Game Features: Normalized temporal and score-related features are incorporated, including time remaining, score margin, event tempo, and possession progress.

(vii) Event History Encoding.: A fixed-length history of recent event types preceding the current action is embedded to capture short-term temporal dependencies.

(vii) Data Sanitization: All features are sanitized to prevent numerical instability, and graph snapshots with inconsistent dimensions are discarded.

(ix) Supervision Structure: Each event is represented as an independent graph snapshot, enabling event-level supervision while preserving the underlying multi-agent structure

2.3 Label Space

In the original NBA play-by-play dataset, events are mapped to *13* distinct action labels, including made field goal, missed shot, turnover, timeout, jump ball, etc. During graph construction and next-action labeling, the jumpball class was systematically removed. Although jump balls can occur during live play (e.g., held-ball situations), they do not appear as next-action prediction targets in our dataset. This is because jump balls function primarily as possession-resolution events rather than strategic decisions and are absorbed into the possession initialization and segmentation logic.

As a result, jump balls almost never constitute the next labeled action following a modeled game state. The "other" category in our label set captures infrequent, administrative, or ambiguous events that appear in the play-by-play but do not correspond to a well-defined basketball decision. Ultimately, the final prediction task uses *12* action labels, focusing only on events that meaningfully represent player or team decision-making.

2.4 Graph Construction

During graph construction, each play-by-play event within a possession is represented as an event-level interaction graph capturing the on-court configuration and game context at that moment. For each event, we construct a graph $G_t = (V, E)$where the node set Vcontains exactly ten nodes, corresponding to the ten players on the court (5 offensive and 5 defensive), identified via reconstructed lineups from starting lineups and substitution logs.

We adopt a player-only graph design because players are the primary decision-making agents in basketball and are consistently identifiable in play-by-play data, whereas other potentially useful entities (e.g., the ball, spatial tracking states, or explicit defender assignments) are not reliably available in the dataset. Although richer heterogeneous graph representations are possible (e.g., including event nodes, ball nodes, or possession-level state nodes), this design focuses on the most stable and directly observable multi-agent structure.

Edges are added between all pairs of players, producing a fully connected interaction graph. This choice supports message passing among all teammates and opponents without requiring incomplete spatial or matchup annotations and allows the GAT to learn which player-to-player interactions are most relevant through attention weights. Events with missing or unreliable on-court lineup information are excluded to maintain consistency and data quality.

2.5 Node and Edge Features

Because player identity is a high-cardinality categorical variable, we represent each player using a learned embedding vector rather than one-hot encoding. This choice allows the model to learn compact representations of individual tendencies and roles across game contexts and is consistent with prior sports prediction work that combines entity embeddings with attention-based relational modeling [Zhang, 2020; Luo & Krishnamurthy, 2023]. To incorporate both identity and context, the final node feature vector is formed by concatenating learned embeddings with engineered contextual features, including role indicators, offense–defense status, normalized game context variables, and short-term event history. One-hot features are cast to floating-point tensors, and numeric variables are normalized prior to concatenation to ensure stable training.

Each node represents a player on the court, and is associated with a feature vector that combines player embeddings and contextual signals. Specifically, each node consists of:

- A 128-dimensional learned player embedding, capturing player-specific tendencies
- A 3-dimensional role indicator denoting whether the player is directly involved in the current event
- A 2-dimensional offense/defense indicator, specifying whether the player is on the offensive or defensive team for the current possession.
- A 6-dimensional game context feature which includes time remaining, score margin, and momentum-related signals derived from the previous event.
- An 8-dimensional learned event-type embedding, so that the model can learn that some events are closer to others (like a missed field goal and a made field goal are both shot attempts, and a foul and a free throw are closely related)
- A 9-dimensional event history that encodes the 3 most recent sequence of event types within the possession.
- A 1-dimensional tempo feature, providing a lightweight proxy for game flow and allowing the model to distinguish between actions occurring in rapid succession versus those following extended stoppages.
- A 1-dimensional possession progress vector that indicates the relative progression of the possession.
- A 3-dimensional rolling-form vector that captures recent shooting efficiency and usage
- A 1-dimensional player load feature, which tries to approximate how tired the player is given the minutes that the player has played since their last substitution.

All features are concatenated into a fixed-length node representation and normalized to avoid numerical instability.

Edges connect all pairs of players on the court. In this project, edges do not carry explicit features; instead, the Graph Attention Network learns weighted interactions implicitly through attention coefficients, allowing the model to adaptively emphasize relevant player-to-player relationships.

2.6 Graph Attention Network

Each interaction graph is processed using a multi-layer Graph Attention Network (GAT). The GAT architecture applies masks to self-attention over neighboring nodes, enabling each player representation to aggregate information from all other players on the court while learning which interactions are most relevant for the current game state

The model uses three stacked GAT layers, each followed by layer normalization and a non-linear activation function. Multi-head attention is employed to stabilize training and capture diverse patterns of player-to-player interaction. After message passing, node embeddings are aggregated via global mean pooling to produce a fixed-dimensional graph-level representation.

This pooled embedding is passed through a feedforward classification head to predict the next action in the possession. Training is performed using cross-entropy loss over a 12-class action space, with gradient clipping and learning-rate scheduling applied to ensure stable optimization.

Overall, the architecture encodes strong relational inductive biases, allowing the model to reason about coordinated multi-agent behavior and structured player interactions

2.7 LLM Baseline

To contextualize GAT's performance, we introduce a Large Language Model (LLM) baseline that operates directly on textual play descriptions. The LLM is prompted with a short natural-language summary of the current game state—including time remaining, score margin, and the most recent play-by-play entry—and is asked to predict the next action from the same label set used by the GAT model. We evaluate both zero-shot and few-shot prompting strategies. In the zero-shot setting, the model receives only task instructions and the current play description. In the few-shot setting, the prompt additionally includes a small number of labeled example sequences to guide the model's predictions. The LLM's textual outputs are then parsed and mapped to the predefined action labels, enabling direct comparison with the GAT's predictions. This baseline serves as a lightweight, unstructured alternative that does not explicitly model player identities, interactions, or spatial relationships. Comparing its performance to the GAT highlights the advantages of explicit relational modeling for fine-grained decision prediction in multi-agent sports environments.

2.8 Training and Optimization

For training and optimization, the model uses cross-entropy loss over the action label space. Training initially uses an unweighted cross-entropy objective with label smoothing. To improve performance on rare action classes, a class-weighted cross-entropy variant is later introduced, but without label smoothing to avoid conflicting regularization effects.

Training is performed with the Adam optimizer with decoupled weight decay. To improve stability during early training, a linear warm-up schedule is applied over the first several epochs, followed by cosine annealing for the remainder of training. This schedule helps mitigate optimization instability caused by large initial gradients in deep attention layers.

Mini-batches of graphs were constructed using PyTorch Geometric's batching mechanism. Gradients are clipped to a fixed maximum norm to prevent exploding gradients, which is particularly important when stacking multiple graph attention layers. When supported by Google Colab's GPU, mixed-precision training is used to accelerate convergence and reduce memory consumption without compromising numerical stability.

Model performance is monitored on a held-out validation set at the end of each epoch. Early stopping is applied based on validation accuracy, and the checkpoint with the best validation performance is selected for final evaluation. All reported metrics are computed using this best-performing model.

2.9 Dataset Splits and Evaluation

The dataset is split at the possession level, assigning all graph instances derived from the same possession to the same split. This choice aligns with the model design: each graph represents a self-contained snapshot that does not depend on future possessions or long-horizon game context. A game-level split would further prevent player and team overlap across splits; however, the focus here is event-level decision modeling rather than generalization across entire games.

Possession-level splitting therefore evaluates the model's ability to make conditional, event-level predictions while avoiding temporal leakage.

3 Experiments and Results

In this section, we evaluate the proposed Graph Attention Network (GAT) for next-action prediction in basketball possessions and compare its performance to a zero-shot Large Language Model (Mistral 7B) baseline. The overall performance is reported per-class performance is analyzed using confusion matrices.

Overall Performance is assessed using Top-1 accuracy, Top-3 accuracy, and macro-averaged F1 score. Top-1 accuracy measures the proportion of graphs for which the highest-probability prediction matches the ground-truth next action. Top-3 accuracy evaluates whether the correct label appears among the three most probable predictions, offering a more permissive measure of predictive usefulness in ambiguous game contexts. Macro F1 accounts for class imbalance by weighing all action classes equally.

Beyond aggregate metrics, per-class performance is analyzed through confusion matrices, which capture class-specific precision and recall behavior across both frequent and rare actions. All results correspond to the best-performing model checkpoint selected based on validation accuracy. For comparison, both the proposed Graph Attention Network and the zero-shot Large Language Model baseline are evaluated under the same protocol and action label space.

3.1 Overall Performance

The Graph Attention Network (GAT) achieves the strongest overall performance, obtaining the highest Top-1 accuracy, Top-3 accuracy, and macro-averaged F1 score. These results indicate that explicitly modeling player interactions and game structure enables more effective next-action prediction than relying on LLMs alone. Incorporating relational inductive biases appears essential for capturing the dynamics of multi-agent basketball environments. Across loss formulations, the GAT trained with label smoothing attains the highest overall accuracy, while the class-weighted cross-entropy variant yields slightly lower Top-1 accuracy but a higher macro-averaged F1 score. This tradeoff reflects improved performance on infrequent event classes—such as ejections and timeouts—which, although not strictly on-ball actions, remain meaningful next-event outcomes. Class weighting therefore helps mitigate class imbalance by encouraging the model to allocate more capacity to rare actions.

In contrast, the LLM baselines exhibit substantially lower accuracy and macro-F1 scores, demonstrating difficulty in predicting fine-grained next actions from textual play descriptions alone. Few-shot prompting provides modest improvements over the zero-shot setting, but the gains remain limited and fall well short of the structured GAT models. Overall, these findings highlight the limitations of unstructured sequence models for event-level prediction and reinforce the value of explicit relational modeling.

3.2 Per-Class Performance

To contextualize model performance, we include a simple frequency-based baseline that always predicts the most common next-action label. Because it ignores player identities, contextual cues, and interaction structure, this baseline quantifies how much predictive power can be attributed solely to class imbalance.

A per-class analysis reveals clear differences in how each model handles frequent versus rare basketball events (Appendix Figures A1–A4). The GAT trained with class-weighted loss (Figure A2) improves recall for structurally constrained and low-frequency actions—such as timeouts, substitutions, violations, and ejections—by redistributing probability mass away from dominant classes. This prevents the model from collapsing onto the most common outcomes and yields a more balanced error profile.

The label-smoothed GAT (Figure A1), by contrast, achieves the highest overall Top-1 and Top-3 accuracy due to strong discrimination among high-frequency on-court actions (made shots, missed shots, rebounds, free throws). However, smoothing reduces sensitivity to rare events, leading to systematic misclassification of administrative or off-ball actions as more common possession-level events.

The LLM baselines (Figures A3–A4) exhibit a qualitatively different pattern. Zero-shot prompting shows a strong tendency to predict the “other” category and struggles to separate fine-grained basketball actions. Few-shot prompting modestly improves recall for majority classes but does not meaningfully improve macro-F1, and the LLM rarely predicts rare actions such as violations or ejections even when contextual cues are explicit.

Model	Top 1	Top 3	Macro F1
Majority-class baseline	0.2400	-	-
GAT (smoothing)	0.5198	0.8671	0.3934
GAT (weighted class)	0.4587	0.8049	0.3948
Mistral (Zero Shot)	0.0786	0.0856	0.0985
Mistral (Few Shot)	0.0779	0.1133	0.0462

Table 1: Comparison of next-action prediction performance across GAT models trained with class-weighted loss and cross-entropy with label smoothing, alongside zero-shot and few-shot LLM baselines. A frequency-based baseline model is also included for reference. Metrics reported include Top-1 accuracy, Top-3 accuracy, and macro-averaged F1.

4 Discussion

The performance differences across models can be understood through the lens of inductive bias, representation structure, and class imbalance. While all models operate over the same action label space, they differ fundamentally in how they encode game state and how they allocate probability mass across frequent and rare events.

4.1 Inductive Bias vs. Language Modeling

The central finding of this work is the clear advantage of models that encode explicit relational structure. The GAT consistently outperforms both zero-shot and few-shot LLM baselines, demonstrating the value of inductive bias for next-action prediction. By representing each possession as a graph over the ten players on the floor, the GAT conditions its predictions on player identity, team membership, offensive and defensive roles, and recent interaction patterns. This structured representation allows the model to reason about who is involved in the play and how prior actions constrain the set of plausible next events.

LLMs, in contrast, operate over linearized play-by-play text that compresses these dependencies into a sequence of tokens. Although the text captures narrative flow, it does not explicitly encode lineup configuration, possession boundaries, or role-specific interactions. As a result, LLMs struggle with state-dependent or rule-driven events—such as substitutions, free-throw sequences, and possession changes—that require precise modeling of game transitions. These limitations highlight that next-action prediction in basketball is fundamentally a relational reasoning task, and that models with explicit multi-agent structure hold a decisive advantage over language-only approaches.

4.2 Rare Events and Class Imbalance

The comparison between training objectives reveals meaningful tradeoffs in how the GAT handles class imbalance. Class-weighted cross-entropy improves macro-F1 by increasing recall for rare but structurally important actions such as ejections, timeouts, and violations. Reweighting encourages the model to devote capacity to these sparse outcomes rather than collapsing into majority classes.

Label smoothing, by contrast, improves overall Top-1 accuracy by tempering overconfidence on frequent actions like made shots, missed shots, and rebounds. However, this comes at the cost of rare-event sensitivity: the smoothed model more often misclassifies administrative or off-ball events as common possession-level actions. These patterns reflect the inherent difficulty of predicting events that are weakly signaled by player interactions. While shots and rebounds have strong relational cues, events like timeouts or ejections depend on external factors such as coaching decisions or officiating. The improved performance of the weighted GAT suggests that combining explicit structural modeling with targeted loss weighting can partially offset this sparsity and enhance recognition of rare but meaningful actions.

4.3 What the Model Is Actually Learning

The GAT model is not learning which player is most likely to take the next shot. Rather, it captures higher-level patterns of game flow conditioned on the multi-agent layout on the court. This includes identifying the current possessions, anticipating administrative transitions, and modeling how recent events constrain the set of plausible next actions. Contextual features such as score margin, time remaining, and possession progress further allow the model to adapt its predictions to situational factors that shape decision-making.

The contrast between administrative and on-court actions provides additional insight into the model's behavior. On-court actions—shots, rebounds, turnovers—are tightly coupled to player interactions and spatial relationships, and the GAT predicts them with relatively high accuracy. Administrative actions such as timeouts, substitutions, and ejections are less directly observable from player dynamics, yet they still benefit from explicit graph structure and class-weighted training. The model's ability to recover these events, despite their weak on-court signatures, highlights the value of incorporating relational and contextual cues.

Overall, the GAT demonstrates an ability to reason about basketball as a coordinated multi-agent system rather than a sequence of isolated events. By attending over player embeddings and interaction context, the model captures how collective behavior evolves throughout a possession. These findings reinforce the importance of relational inductive biases for modeling complex, structured decision processes in sports environments.

Limitations

While the proposed graph-based framework demonstrates strong performance on next-action prediction in basketball, several limitations remain. The approach depends heavily on the accuracy and consistency of NBA play-by-play annotations, which can contain ambiguities, season-to-season inconsistencies, and occasional missing or delayed events. Administrative actions such as ejections or timeouts are only loosely connected to on-court interactions, meaning that errors or omissions in the underlying logs propagate directly into graph construction and labeling.

A second limitation arises from the heuristic reconstruction of on-court lineups and possessions. Lineups are inferred from substitution patterns and early-game participation, a procedure that works well for most possessions but can fail in edge cases involving incomplete substitution records or corrupted game segments. Possessions for which lineups cannot be reliably reconstructed are excluded, introducing a mild selection bias toward cleaner, more complete data.

The model also lacks explicit spatial information. Because it operates solely on play-by-play text rather than player-tracking data, the GAT reasons about interactions and roles abstractly rather than geometrically. This design choice enables broad applicability but limits the model's ability to capture fine-grained tactical patterns such as defensive spacing, shot contesting, or off-ball movement.

Class imbalance further poses challenges. Even with weighted loss functions, rare events such as ejections and violations remain difficult to predict, in part because they depend on contextual factors external to player interactions, including officiating decisions. Although class weighting improves recall for these events, overall performance remains limited, underscoring the inherent difficulty of modeling low-frequency administrative actions.

Finally, the LLM baseline is intentionally simplified. It relies only on zero-shot and few-shot prompting of an off-the-shelf model, without fine-tuning, retrieval augmentation, or structured decoding. As a result, the reported LLM performance should be interpreted as a lower bound on what language-based approaches might achieve with more extensive adaptation. Moreover, although the dataset spans multiple seasons, evaluation is conducted on held-out games rather than entirely unseen seasons or rule regimes, leaving open questions about generalization across shifts in league style, officiating emphasis, or roster composition.

Conclusion and Future Work

This study demonstrates that next-action prediction in basketball benefits substantially from structured, interaction-aware modeling. By encoding each possession as a graph over the ten on-court players and incorporating contextual game information, the Graph Attention Network captures relational dependencies and short-term game flow that remain inaccessible to sequence-only language models. The resulting performance gains, particularly on structurally constrained and infrequent actions, highlight the importance of relational inductive biases for event-level prediction in multi-agent sports environments.

Future research may extend this framework by evaluating generalization across unseen seasons, integrating spatial tracking data to capture fine-grained tactical behavior, or exploring hybrid architectures that combine graph-based representations with pretrained language models. Such directions offer promising avenues for advancing predictive modeling, simulation, and decision-support tools within sports analytics.

Acknowledgements

I want to thank Professor Zhou Zhang for his guidance and feedback on this project.

References

Li, Junjie and Jiang Wei. (2025). Analysis of basketball tactical intent recognition system with collaborative graph convolutional network and spatiotemporal attention mechanism. AKCE International Journal of Graphs and Combinatorics. https://doi.org/10.1080/09728600.2025.2552178

Luo, Rui and Vikram Krishnamurthy. (2023). Who you play affects how you play: Predicting sports performance using graph attention networks with temporal convolution. arXiv preprint arXiv:2303.16741. https://arxiv.org/abs/2303.16741

Sports-Statistics.com. (n.d.). NBA Basketball Datasets (CSV Files). Retrieved February 2, 2026, from https://sports-statistics.com/sports-data/nba-basketball-datasets-csv-files/

Wu, Yen Tsang, Jenq Haur Wang, and Ning Chien. (2021). Predicting the outcome of games based on graph neural networks. Camera ready manuscript, National Taipei University of Technology.

Zhang, Tony. (2020). Graphsketball: NBA meets graph ML. Medium / Stanford CS224W Blog. https://medium.com/stanford-cs224w/graphsketball-nba-meets-graph-ml-cf5b67b7d46c

Zhao, Kai, Guangxin Tan, Zehong Chen, Jinluo Axi, and Qian Huang. (2025). Improving basketball prediction performance with graph attention networks and K-means clustering. Proceedings of the 2024 International Conference on Sports Technology and Performance Analysis (ICSTPA '24). Association for Computing Machinery, New York, NY, USA, 227–234. https://doi.org/10.1145/3723936.3723970

Appendix

Figure A1. Confusion Matrix for GAT with Label Smoothing

Figure A2. Confusion Matrix for GAT with Class Weights

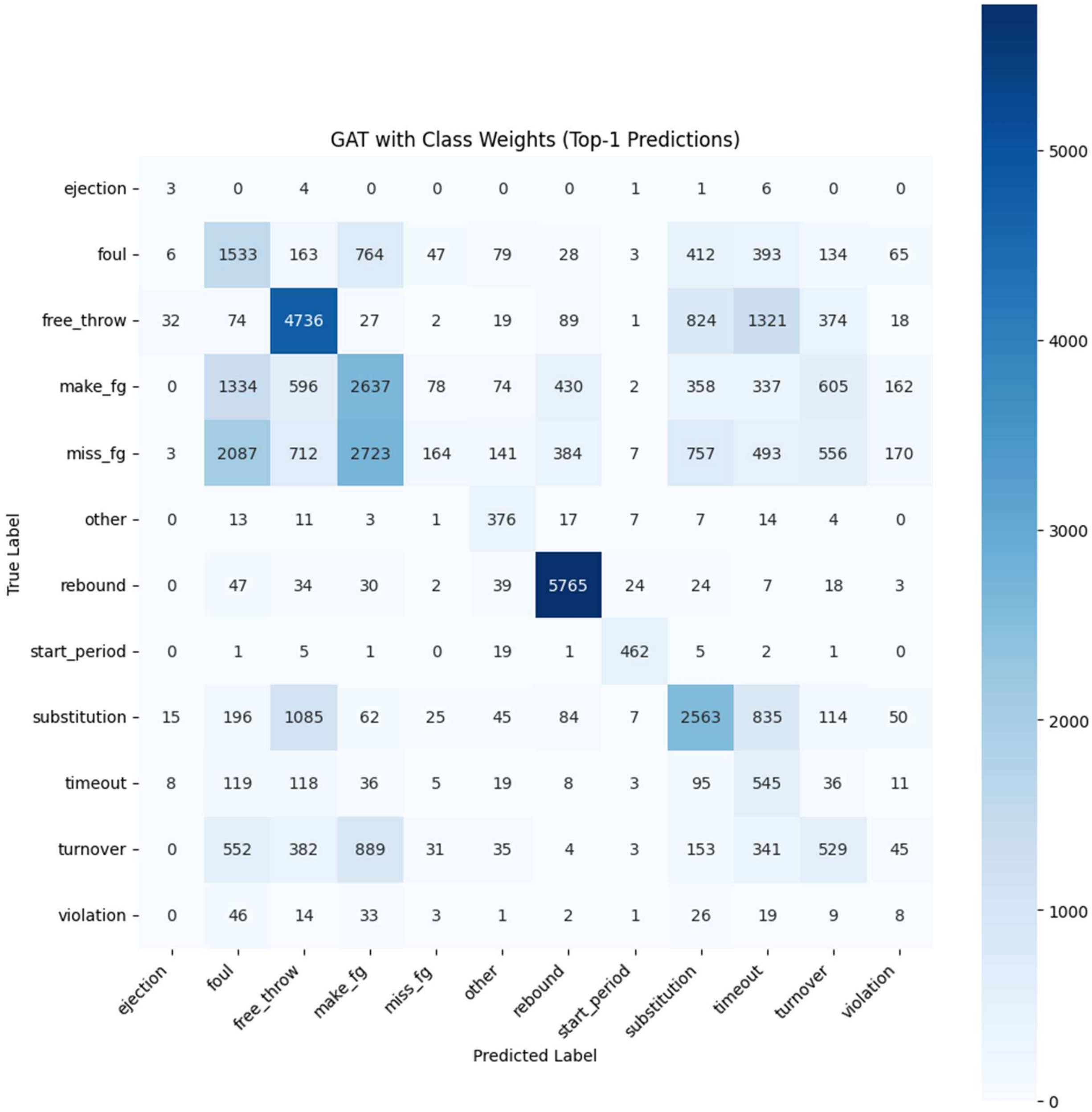

Figure A3: Confusion Matrix for Mistral 7B Zero-Shot

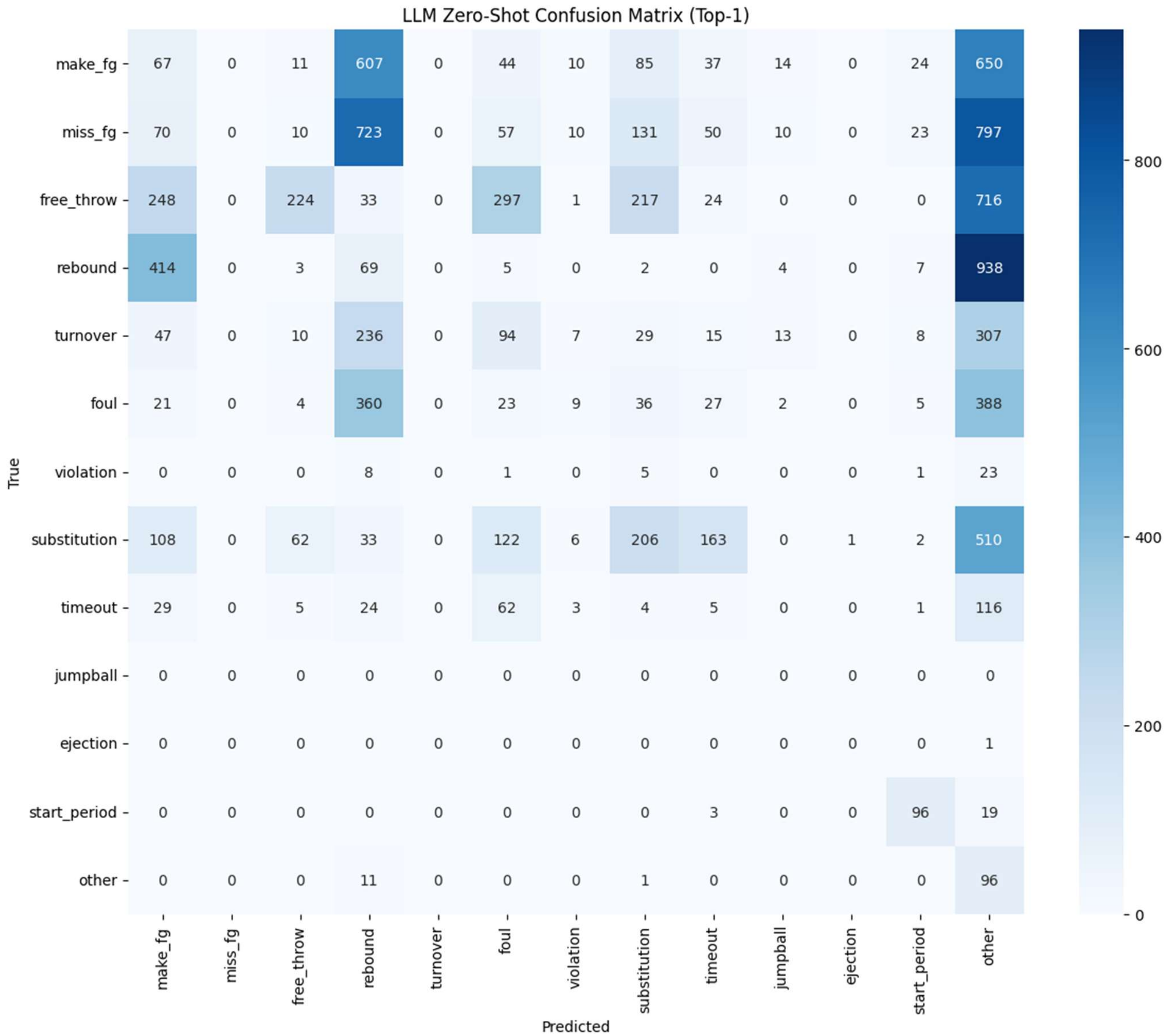

Figure A4: Confusion Matrix for Mistral 7B (Few-Shot)

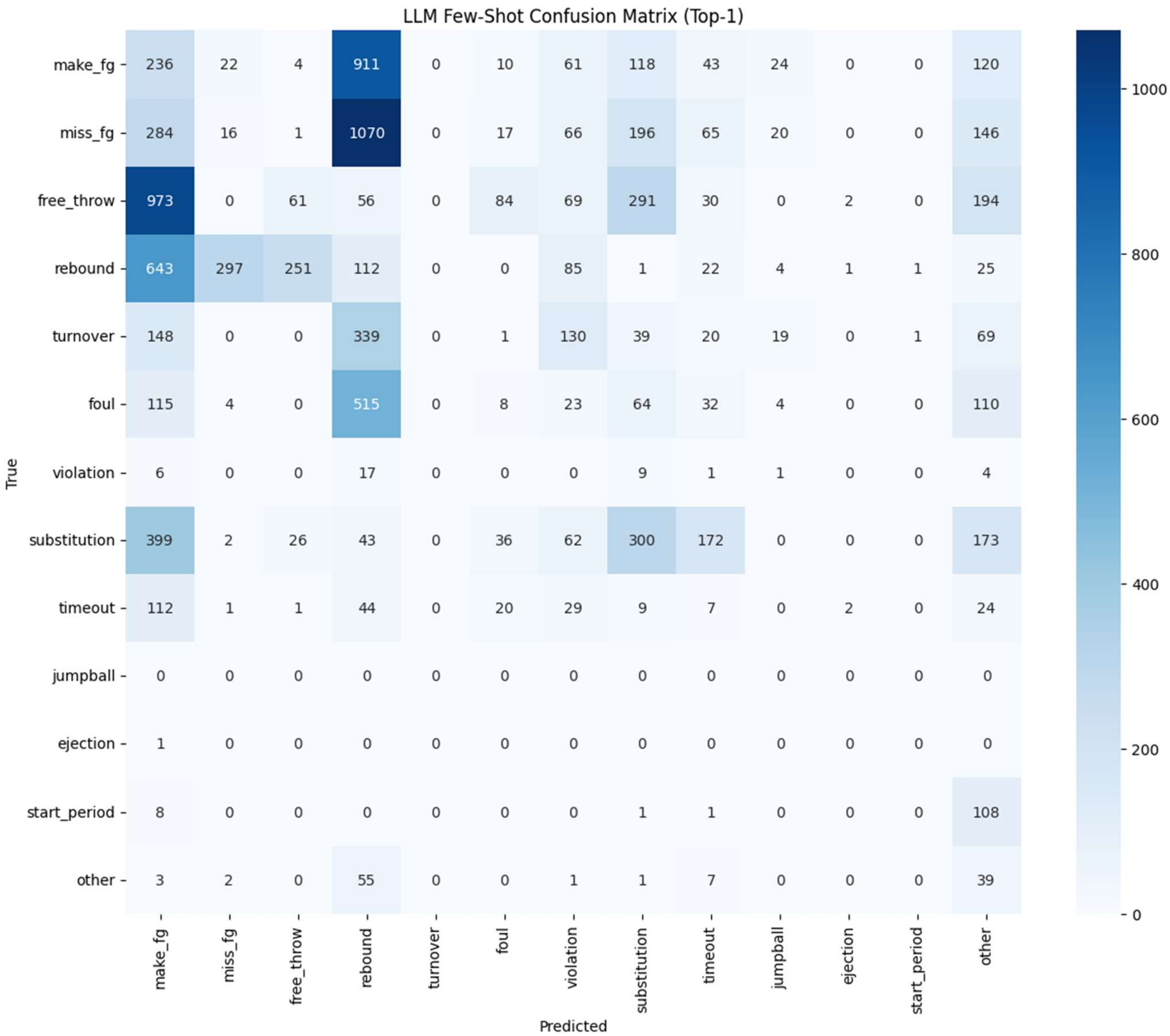

Figure A5: Example Label Space

	HOMEDESCRIPTION	VISITORDESCRIPTION	NEUTRALDESCRIPTION	EVENTMSGTYPE	EVENTMSGACTIONTYPE	SCORE	POSSESSION_ID
0	NaN	NaN	NaN	12	0	NaN	1
1	Jump Ball Camby vs. Ratliff: Tip to Houston	NaN	NaN	10	0	NaN	2
2	MISS Sprewell 6' Jump Shot	Ratliff BLOCK (1 BLK)	NaN	2	1	NaN	2
3	NaN	76ers Rebound	NaN	4	0	NaN	3
4	Camby S.FOUL (P1.T1)	NaN	NaN	6	2	NaN	3
5	NaN	Ratliff Free Throw 1 of 2 (1 PTS)	NaN	3	11	1 - 0	3
6	NaN	MISS Ratliff Free Throw 2 of 2	NaN	3	12	NaN	3
7	Ward REBOUND (Off:0 Def:1)	NaN	NaN	4	0	NaN	4
8	NaN	Ratliff S.FOUL (P1.T1)	NaN	6	2	NaN	4
9	Camby Free Throw 1 of 2 (1 PTS)	NaN	NaN	3	11	1 - 1	4
10	Camby Free Throw 2 of 2 (2 PTS)	NaN	NaN	3	12	1 - 2	4
11	NaN	MISS Iverson 20' Jump Shot	NaN	2	1	NaN	5
12	Camby REBOUND (Off:0 Def:1)	NaN	NaN	4	0	NaN	6
13	MISS Houston 15' Jump Shot	NaN	NaN	2	1	NaN	6
14	Ward REBOUND (Off:1 Def:1)	NaN	NaN	4	0	NaN	6
15	Ward Bad Pass Turnover (P1.T1)	NaN	NaN	5	1	NaN	6
16	NaN	MISS Hill 10' Jump Shot	NaN	2	1	NaN	7
17	Sprewell REBOUND (Off:0 Def:1)	NaN	NaN	4	0	NaN	8
18	MISS Sprewell 14' Jump Shot	NaN	NaN	2	1	NaN	8
19	NaN	Lynch REBOUND (Off:0 Def:1)	NaN	4	0	NaN	9
20	NaN	Ratliff Layup (3 PTS) (Lynch 1 AST)	NaN	1	5	3 - 2	9
21	Houston 16' Jump Shot (2 PTS) (Ward 1 AST)	NaN	NaN	1	1	3 - 4	10
22	NaN	Ratliff Traveling Turnover (P1.T1)	NaN	5	4	NaN	11
23	NaN	Snow P.FOUL (P1.T2)	NaN	6	1	NaN	12
24	MISS Johnson Layup	Hill BLOCK (1 BLK)	NaN	2	5	NaN	12
25	NaN	76ers Rebound	NaN	4	0	NaN	13
26	Johnson L.B.FOUL (P1.T2)	NaN	NaN	6	3	NaN	13
27	NaN	Ratliff Slam Dunk (5 PTS) (Lynch 2 AST)	NaN	1	8	5 - 4	13
28	MISS Camby 17' Jump Shot	NaN	NaN	2	1	NaN	14
29	NaN	Iverson REBOUND (Off:0 Def:1)	NaN	4	0	NaN	15