DeepPlaybook: Deep Learning-Based Basketball Play-by-Play Analysis

Mike Boss
Eidgenössische Techn.
Hochschule Zürich

Konstantin Heep
Eidgenössische Techn.
Hochschule Zürich

Romeo Stoll
Eidgenössische Techn.
Hochschule Zürich

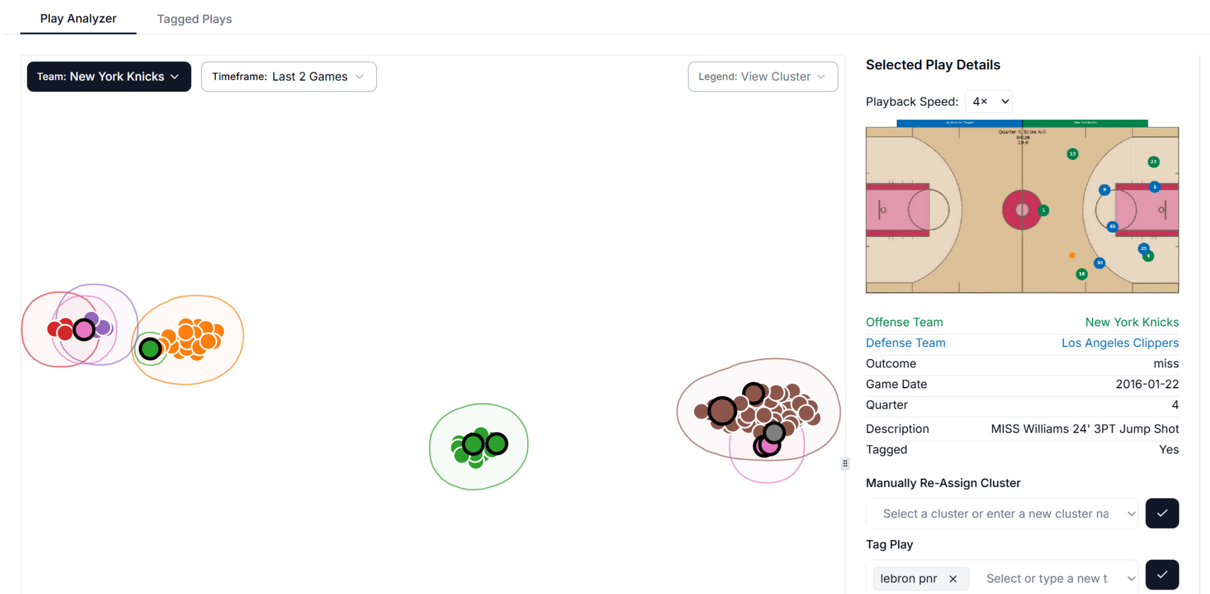

Figure 1: Main view of the interactive dashboard.

Abstract

The task of scouting an upcoming opponent in professional Basketball often requires hours of manual labor, with only a small proportion of it requiring the domain expertise of the coaches carrying out the task. Yet the task is not easily solved by an algorithm or an ML agent. We propose the mixed-initiative system *DeepPlaybook* that enables easy exploration, organization and presentation of Basketball plays in an interactive dashboard with a semi-supervised projection and an interactive clustering fueled by a transformer model with a VAE bottleneck.

Index Terms: Basketball, Play-by-play analysis, interactive dashboard, semi-supervised projection.

1 Introduction

Plays are an important strategic part of professional basketball. Although points can be won by overwhelming the opposing team with speed, and in 1v1 situations, when the defense is in place, offensive plays can be used to gain an edge. The different plays the coaches implement are called *playbooks*. Especially in the NBA, playbooks often contain only a handful of plays and change frequently. To prepare for a match, teams, therefore, look closely at the last games the opponent played. In addition to which play is used how frequently, they also try to find out how the play is initiated, by which player, and in which role. In practice, an assistant coach will go through the recordings of the last two to five games of the opponent team, manually cut the videos into plays, analyze them, and prepare a presentation for the team and head coach. This task can take hours and requires little basketball expertise for the most part. Our work addresses this problem by providing *DeepPlaybook*, an interactive dashboard for basketball play-by-play analysis. We implemented the dashboard for the NBA Player Movements[2] dataset. In section 2 we explain and motivate different parts of the interface and give a description of the workflow. In section 3 we explain the model training. In section 4 we explain the semi-supervised UMAP projection and interactive clustering. In section 5 we summarize our contributions and explore some future work.

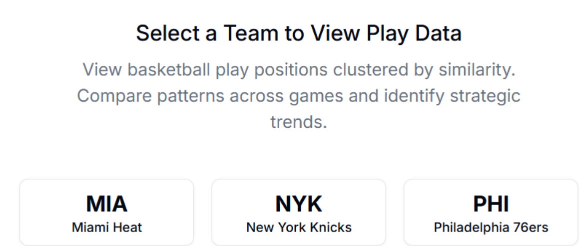

Figure 2: The *DeepPlaybook* landing page lets the user choose from several teams. In the current implementation the number of teams is limited to three due to limited resources.

2 The *DeepPlaybook* Interface and an Exemplary Workflow

The *DeepPlaybook* interface is optimized to give a smooth workflow for an assistant coach scouting an opponent team. In this section, we present an exemplary workflow. The landing page allows the user to choose a team to scout (Figure 2).

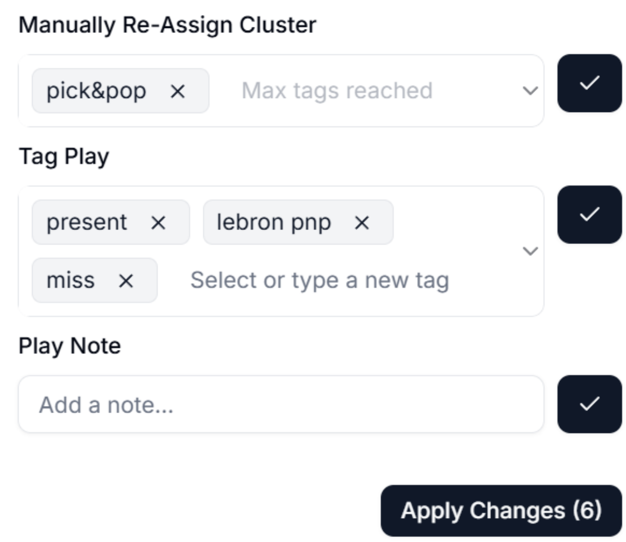

Figure 3: Closeup view of the fields for cluster re-assignment and play tags.

Upon selection of a team, the user is presented with the main view of the dashboard (Figure 1) with a scatter plot visualization of all plays of the last games played by the opponent team. An interactive component lets the user choose the number of games to consider. The visualization allows direct manipulation of the plot by panning and zooming, getting additional information by mouse hover or selection of points. Based on learned embeddings, the plays are projected into 2D space with UMAP and clustered by similarity. A dropdown menu let's the user re-center the view on any cluster. By allowing the user to re-assign points to different or new clusters, the user can adapt and personalize the projection into 2D space. They are encouraged to initially select several points from each cluster and manually assign them in an intuitive way for them. For example, stationary and strategical set plays can be separated from more dynamic fast breaks or different plays involving screens can be arbitrarily broken up into several clusters. Just reassigning the points does not change their position in the plot until the ”Apply Changes” button is pressed. This sends the assignments to the backend and creates a new projection based on the input. The adapted clustering aligns well with the user's categories and therefore makes navigation and exploration easier. Upon selecting a point, details about the play and the cluster are displayed on a panel on the right side of the screen. Most importantly it contains information for this play about possession (is the team in question in offense or in defense?), outcome (was the play a miss or did it score points?), as well as information about the cluster (how frequently is this play used and how successful is it on average?). An animation of the play with controls to play / pause, skip forward and backward, and to select a playback speed let the user inspect the play in detail. The target user is deeply familiar with basketball plays and can pick up a lot of information quickly. Therefore, viewing the video even on a high playback speed allows them to gain more information than any feasible classifier could extract. The playback speed is kept consist once selected and can be adapted for quick inspection, in-depth analysis or presentation. Once the clustering has been refined, a list of similar plays, displayed at the bottom of the page, allows the user to quickly go through other plays in a cluster. A function to set tags and add notes to plays allows the user to keep track of findings throughout analysis. A second tab lets them view

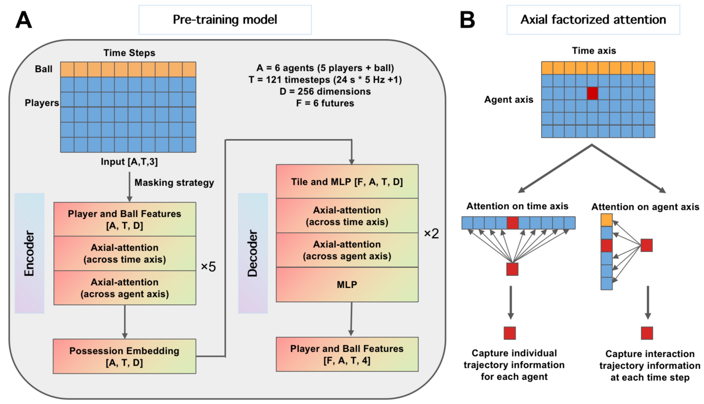

Figure 4: The HoopTransformer architecture described in [5]. The architecture features a transformer-based encoder and decoder, with the possession embeddings as the hidden state in between. It features two different types of attention, across time and across agent attention. The model is trained by predicting masked inputs. Both the time and agent dimension are input in a batched fashion, while the attention operates on each at a time.

tagged plays by category. This is especially useful to eventually present the results to a head coach or strategy team. The whole process is highly efficient: From a quick (optional) personalization of the overview, the user can jump right into exploration and analysis with features guided by the suggested similarities and separation of points. At the same time it encourages the user to easily keep track of and prepare analysis results for presentation.

3 Model Training

We initially explored custom feature engineering with a Transformer architecture, but switched to adopting the HoopTransformer model [5] while retaining key preprocessing components from our original approach.

3.1 Dataset and Preprocessing

Using SportVU tracking data from 632 NBA games (2015-2016 season), we filtered the dataset from ∼280,000 to 90,524 possessions (plays). Each possession represents a naturally segmented offensive sequence (≤24s per basketball rules) ending in a shot attempt. To focus on offensive tactical patterns and reduce computational overhead, we excluded defensive player tracking data, retaining only ball and offensive player movements. We also filter out full-court transitions, keeping only half-court offensive plays, and normalized the court orientation so that the offense consistently moved in the same direction.

3.2 Architecture

HoopTransformer employs attention-based multi-agent motion prediction, adapted from autonomous vehicle trajectory modeling [5]. The model learns offensive tactical representations through masked motion prediction tasks, enabling it to reconstruct player trajectories while capturing underlying strategic patterns for set play recognition. The complete architecture can be seen in 4

3.3 Training Process

Training was carried out for 50 epochs using an 80/20 train-validation split. Following the HoopTransformer approach [5], we employed a joint masking strategy combining motion prediction (MP) and motion reconstruction (MR) tasks. This joint MP + MR strategy enables the model to both predict future player

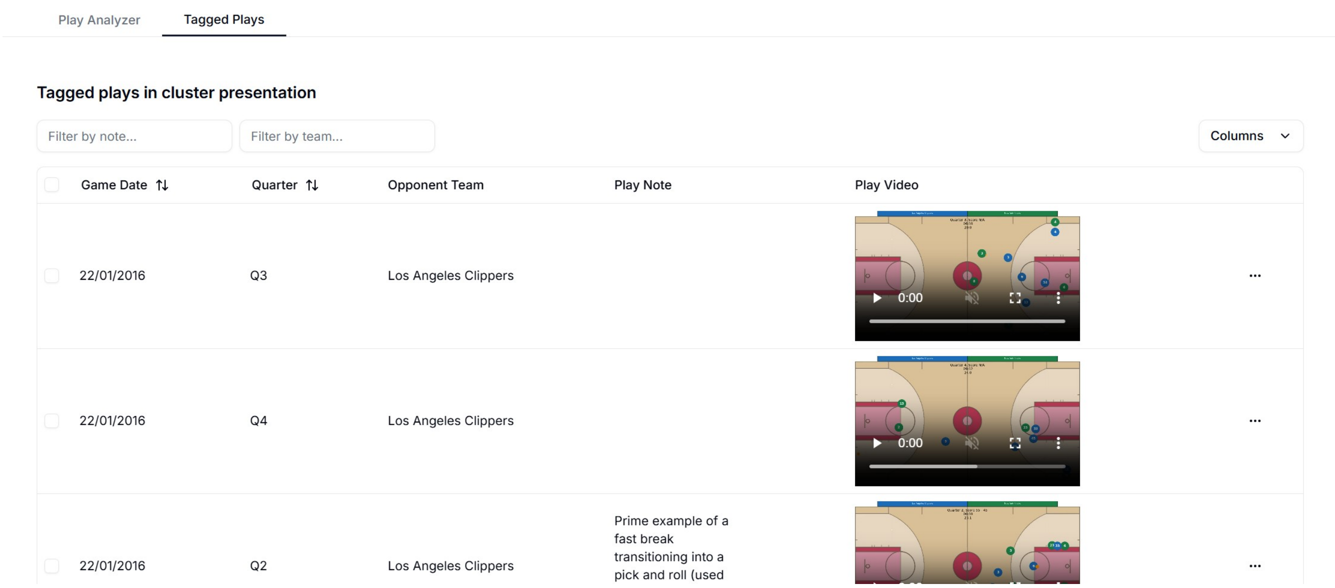

Figure 5: The *Tagged Plays* view of the dashboard allows a user to quickly go through the tagged plays for hypothesis generation, evaluation or presentation.

movements and reconstruct masked historical trajectories, providing richer learning signals for tactical pattern recognition.

Rather than optimizing purely for trajectory reconstruction, we extracted the trained encoder to generate 90,524 possession embeddings. These embeddings are the foundation for similarity search and k-nearest neighbor clustering to identify tactical patterns across possessions.

This approach allows the model to learn tactical representations that generalize beyond individual trajectory prediction to a broader strategic understanding.

4 Supervised Projection and interactive clustering

Our system uses a semi-supervised UMAP [3] projection to enable interactive clustering without the need to recompute the underlying transformer embeddings, allowing for efficient interactive updates to the play visualization.

4.1 Initial Clustering and Projection

To create the initial projection, the embeddings are clustered once in high dimensions using K-Means from faiss [1] using a fixed number of clusters. As no user preferences are available initially, the K-Means clusters are directly used to supervise UMAP for the initial projection. This full supervision will separate clusters in the projection while keeping intra-cluster structures to analyze.

4.2 Semi-Supervised Re-projection

Users can refine the clustering through interaction with the scatter plot or similar play interface. The user can either assign a different cluster or create a new cluster for each play. When the user creates new clusters, the system automatically populates them with similar untagged plays using nearest-neighbor search in the embedding space. When users apply their changes, the system recomputes only the UMAP projection using updated cluster assignments as supervision for UMAP, leaving the transformer embeddings unchanged. This allows for fast interactive updates while preserving the learned relationships by the model.

The system persists user modifications separately from initial clustering, enabling incremental refinement across sessions. This allows users to easily revert changes to their original state.

5 Conclusion and Future Work

While the tool is already highly streamlined, there are several directions in which the project can be advanced, particularly in developing a more sophisticated human-AI collaborative learning framework.

5.1 Co-Adaptive Analytics and Interactive Learning

Our current implementation allows users to manually reassign possessions in the interactive dashboard, triggering supervised UMAP recomputation. When creating new clusters, the system automatically moves the most similar plays to the new cluster while maintaining the cluster stability of the other clusters by mimicking k-means re-clustering behavior. However, this represents only the initial step toward co-adaptive analytics.

Future development should implement a fine-tuning feedback loop in which the model learns from user interactions. Specifically, we propose unfreezing the final two encoder layers and training them concurrently with a linear classification head, using user cluster reassignments as supervision signals. This approach would enable the model to adapt its learned representations based on domain expert feedback, creating a bidirectional learning system where both user understanding and model performance improve iteratively.

The coadaptive framework could be extended with active learning strategies, where the system identifies ambiguous possessions and requests user input on the most informative examples. This would maximize learning efficiency while minimizing annotation burden. Additionally, implementing confidence scoring for model predictions would allow the system to highlight uncertain classifications, directing user attention to cases where expert knowledge is most valuable.

5.2 User Experience and Accessibility

For user guidance, automatic onboarding via intro.js [4] is already in development to ensure feature awareness and correct usage.

Another, more engaged way to make the usage more intuitive would be to separate the optional clustering personalization into a dedicated view. This could make the dashboard more accessible by gradually introducing features and reducing the cognitive load.

5.3 Data Organization and Task Extension

Additional data organization capabilities could enable exploration by temporal dimensions (date, quarter), performance metrics (points scored), or contextual factors. Of course, there are also related analytical tasks relevant to the target user that the dashboard could expand to support.

Regardless of future developments, the tool's strength lies in its focused approach to a clearly defined problem. It represents a co-agentic solution that empowers domain experts to achieve greater impact with reduced effort while maintaining human oversight over complex decisions that cannot be fully automated.

REFERENCES

[1] M. Douze, A. Guzhva, C. Deng, J. Johnson, G. Szilvasy, P.-E. Mazaré, M. Lomeli, L. Hosseini, and H. Jégou. The faiss library. 2024. 3

[2] K. L. [linouk23]. Nba-player-movements. Github, September 2016. 1

[3] L. McInnes, J. Healy, N. Saul, and L. Großberger. UMAP: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3(29):861, 2018. doi: 10.21105/joss.00861 3

[4] Usablica. intro.js, 2025. [Source code], v7.2.0. 3

[5] X. Wang, Z. Tang, J. Shao, et al. Hooptransformer: Advancing nba offensive play recognition with self-supervised learning from player trajectories. *Sports Medicine*, 54:2663–2673, 2024. doi: 10.1007/s40279-024-02030-3 2