ATTENTION IS ALL YOU NEED, FOR SPORTS TRACKING DATA

PREPRINT

Udit Ranasaria
SumerSports
udit.ranasaria@sumersports.com

Pavel Vabishchevich
SumerSports
pavel.vabishchevich@sumersports.com

August 20, 2024

ABSTRACT

The rapid advancement of spatial tracking technologies in sports has led to an unprecedented surge in high-quality, high-volume data across all levels of play. While this data has catalyzed innovations in sports analytics, current methodologies for team sports often struggle with the inherent challenge of the player-ordering problem. This paper highlights the application of Transformer architectures and Attention to address these challenges in sports analytics. Our approach operates end-to-end on raw player tracking data, processes unordered collections of player vectors, and is inherently designed to learn pairwise spatial interactions between players. The framework satisfies critical criteria for widespread adoption in sports modeling: minimal feature engineering, adaptability across diverse problems, and accessibility in terms of understandability and reproducibility. To demonstrate its effectiveness, we apply our approach to the task of predicting tackling location in the NFL, a problem recently explored in the public domain. Our results show significant improvements over commonly used approaches, achieving 20% better prediction accuracy (4.61 vs 5.78 yards average displacement error) and particularly excelling in generalizing to diverse game situations. This work aims to advance sports analytics research methodologies by demonstrating the potential of Transformer-based architectures over traditional models. The potential implications include unlocking new insights into player dynamics, team strategies, and game outcomes across various sports domains, paving the way for more sophisticated deep learning models in sports analytics. [1]

Keywords sports analytics · sports tracking data · latent representation learning · multi-agent spatiotemporal modeling

1 Introduction

The field of sports analytics has experienced rapid growth, driven by unprecedented advancements in spatial tracking technologies. High-quality, high-volume data acquisition, facilitated by optical systems, computer vision algorithms, and chip-based tracking solutions, is now pervasive across all sports and levels of play. This confluence of rich data with innovative research and sophisticated modeling techniques has precipitated disruptions in multiple domains, including sports science, broadcasting, player evaluation, team building optimization, injury prevention, and tactical strategy formulation. Kovalchik [2023] comprehensively summarizes the burgeoning corpus of innovative research contributions leveraging sports tracking data, with the most advanced and modern modeling emphasis in Latent Variable Estimation, Event Prediction, and Value Attribution. Despite the sophistication of these methodologies, their potential is often hampered by reliance on traditional approaches to circumvent the *player-ordering problem*. This challenge arises from the dynamic nature of team sports, where player roles and formations are inconsistent and can vary between games. The absence of a persistent intrinsic order of players conflicts with the input structure requirements of many standard machine learning models. This necessitates hand-crafted feature preprocessing steps to transform raw tracking data into structured feature representations or the imposition of heuristic-based orderings [Horton, 2020]. These hand-designed feature extractors or ordering approaches, while functional, rely on domain expertise and lack generalizability. Moreover, with the proliferation of deep learning, feature engineering has become an anti-pattern with preference

[1] Our code is openly available here

given to *end-to-end learning* [LeCun et al., 2015, Ng, 2018]. Advocates for end-to-end learning emphasize that, given sufficient data, models should aim to learn latent features directly from raw inputs, optimizing against training objectives. Consequently, modeling advances in deep learning research typically stem from architectural innovations tailored to the data space rather than feature extraction techniques. Furthermore, we posit that the majority of modeling tasks involving sports player tracking share a fundamental objective: learning pairwise spatial interactions between players. Models adept at capturing these interactions are likely to excel in event prediction and other supervised tasks that necessitate a nuanced understanding of player positioning within the context of sport-specific rules and dynamics. This observation motivates our exploration of novel architectural approaches that can inherently handle unordered sets of player data while capturing complex spatial relationships. To address these challenges, this paper highlights the application of Transformer architectures as a modeling framework for sports analytics. Transformers, originally developed for natural language processing tasks, have shown remarkable capabilities in handling sequential and unordered data across various domains. Our proposed approach satisfies several critical criteria:

1. End-to-end operation on raw player tracking data with minimal feature engineering, accepting any set of spatiotemporal features (positions, velocities, orientations, player attributes, etc.) available in the tracking system.
2. Capability to process unordered collections of player vectors, directly addressing the player-ordering problem across any team sport.
3. Inherent design for learning pairwise player interactions.
4. Accessibility in terms of understandability, explainability, and reproducibility.

Our objective is to encourage a methodological advancement in sports analytics research. We anticipate a gradual transition from traditional models like XGBoost and MLPs, commonly employed in competitions such as the NFL Big Data Bowl, towards variants and extensions of Transformer architectures. This shift has the potential to unlock new insights in player dynamics, team strategies, and game outcomes, ultimately advancing the field of sports analytics and its applications across various domains.

2 Methods

2.1 Sports Tracking Data

Table 1: Example of a Multi-Agent Tracking Frame from the NFL

frame_id	event	nfl_player_id	team	x	y	s	o	dir
15	handoff	34452	OFF	26.87	27.9	3.0	274.91	242.98
15	handoff	40089	OFF	35.19	34.03	0.51	246.43	84.06
15	handoff	42368	DEF	40.37	25.44	6.15	304.19	297.75
⋮								
15	handoff	54948	DEF	35.33	31.36	2.67	272.22	261.64

Sports tracking data provides rich spatiotemporal information about player movements and game events. This data typically includes:

- A frame number or time column to track the temporal progression of events
- A unique player identifier for each athlete on the field
- An indicator for team affiliation (e.g., offense or defense)
- An event stream that annotates specific "actions" occurring at given moments in the game
- Feature columns representing spatial properties of each player, such as position and velocity

Table 1 presents a sample of data from the 2024 NFL Big Data Bowl, illustrating these key components.

In this paper, we focus on modeling the static *multi-entity* aspect of tracking data that exists within a single frame (or timestamp). We treat each tracking frame at a given timestamp as a unique, independent training sample, disconnected from the frames temporally surrounding it. This approach allows us to focus the discussion around the spatial relationships between players at specific moments, which is crucial for many predictive tasks in sports analytics.

2.2 Task Formulation

To facilitate a comparative discussion of various modeling approaches, we first establish a general mathematical framework for learning from sports tracking data. Let $P = \{p_1, p_2, ..., p_K\}$ represent the set of K players participating in a particular frame. Similarly, let $V = \{v_1, v_2, ..., v_K\}$ be the set of feature vectors such that each $v_k \in \mathbb{R}^d$ captures all relevant spatial (e.g., position and velocity) and characteristic (e.g., height and weight) features for the player p_k. Crucially, both P and V are *unordered* sets, reflecting the absence of intrinsic order of players in most team sports. We define our supervised learning task over V as follows: Let $y \in \mathcal{Y}$ be the objective label we aim to predict, where $\mathcal{Y}$ is the set of possible outcomes. This could represent various tasks such as predicting future events or outcomes. We define a model $f : (\mathbb{R}^d)^K \to \mathcal{Y}$ as a function that maps the set of feature vectors V to the label space $\mathcal{Y}$. Formally,

$$\hat{y} = f(V) = f(\{v_1, v_2, ..., v_K\}) \tag{1}$$

where $\hat{y}$ is the predicted label. To train this model, we define a loss function $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ that measures the discrepancy between the true label y and the predicted label $\hat{y}$. The optimization problem can then be formulated as:

$$f^* = \arg\min_f \mathbb{E}_{(V,y)\sim\mathcal{D}}[\mathcal{L}(y, f(V))] \tag{2}$$

where $\mathcal{D}$ represents the underlying data distribution from which our training samples are drawn. This formulation describes a general framework for modeling on unordered sets of player data. A critical property for such models is **permutation equivariance**:

Definition (Permutation Equivariance and Invariance). A function $f : (\mathbb{R}^d)^K \to (\mathbb{R}^{d'})^K$ is **permutation-equivariant** if for any permutation π of the input indices:

$$f(\{v_{\pi(1)}, v_{\pi(2)}, ..., v_{\pi(K)}\}) = \{f(v)_{\pi(1)}, f(v)_{\pi(2)}, ..., f(v)_{\pi(K)}\}$$

In other words, permuting the input players results in the same permutation applied to the output representations.

For our tackle prediction task, we require **permutation invariance**: the final prediction should be unchanged by player ordering. This is achieved by composing a permutation-equivariant encoder (the Transformer) with a permutation-invariant decoder (pooling + MLP), such that for any permutation π:

$$f(\pi(V)) = f(V) \quad \text{where } f : (\mathbb{R}^d)^K \to \mathcal{Y}$$

In the following sections, we will discuss various prior approaches to ascribing a model to f, highlighting their strengths and limitations in handling the player-ordering problem. This will set the stage for our proposed Transformer-based approach, which we argue offers a more flexible and effective solution to learning from unordered sports tracking data.

2.3 Decomposition Approaches in Prior Work

As discussed in the Introduction 1, often sports researchers identify this player-ordering problem and then decompose the modeling as:

$$f(V) = g(\Phi(V)) \tag{3}$$

where $\Phi : (\mathbb{R}^d)^K \to \mathbb{R}^m$ is a feature extraction or player ordering process that maps the unordered set of player feature vectors to an ordered fixed-dimensional representation, and $g : \mathbb{R}^m \to \mathcal{Y}$ is typically implemented using MLPs or gradient boosted tree models. Applying a fixed ordering over V based on domain heuristics is still considered a special case of Φ where: $\Phi_{\text{fixed}} : (\mathbb{R}^d)^K \to \mathbb{R}^{d \cdot K}$.

Several notable examples in the literature follow this decomposition paradigm:

- Both Fernández et al. [2019] and Yurko et al. [2020] propose novel frameworks to decompose complex sports like soccer and American football into continuous time value-based metrics. Both papers invest heavily in deriving "a wide set of spatio-temporal features" to feed individual models that comprise the frameworks.
- Amirli and Alemdar [2022], in building a model to infer ball location from tracking data, identify that "it is impossible to find a correct ordering for the individual players to be represented in the feature matrix" and implement a segment-based role assignment algorithm to fix an order.
- Le et al. [2017] and Schmid et al. [2021] employ deep imitation learning for "ghosting" and team-strategy evaluation in soccer and American football, respectively. To featurize a tracking frame as input into recurrent nets, they rely on a role-based assignment step to "impose ordering on the training input".
- Felsen et al. [2018] built a conditional variational auto-encoder model capable of synthetically generating basketball player trajectories conditioned on identity and context. They separately develop an algorithm to solve the "significant challenge in encoding multi-agent trajectories is the presence of permutation disorder".

- Mehrasa et al. [2018] innovates with convolutional network filtering over the time dimension but uses an anchor-based sorting scheme to avoid "implicitly enforc[ing] an order among this set of players".

This collection, while not exhaustive, illustrates a common pattern in advanced sports research: encountering the player-ordering problem and addressing it through various feature engineering or ordering schemes. These approaches, while effective for specific tasks, often lack generalizability and may not fully capture the complex spatial relationships inherent in sports tracking data.

Given the trends in deep learning towards end-to-end solutions, we argue that an approach capable of discovering latent features directly from the data, without the need for explicit feature engineering or ordering, could potentially outperform these methods across a wider range of tasks.

2.4 Generalized Transformer Model Architecture

The Transformer architecture, introduced by Vaswani et al. [2017], revolutionized Natural Language Processing by introducing a self-attention mechanism that enables learning from direct pairwise interactions between all elements in a sequence, regardless of their order. This approach effectively addresses the challenge of modeling long-range dependencies, a limitation of commonly used recurrent models.[2] Crucially, the overall Transformer architecture maintains permutation equivariance: any permutation of the input sequence results in the same permutation of the output embeddings, without affecting the values of the embeddings in any way. This property is *exactly* what is needed to directly learn over the unordered set of player feature vectors end-to-end while capturing player interaction relationships. We define our Transformer-based model as:

$$f(V) = g(\text{TransformerEncoder}(V)) \tag{4}$$

where the TransformerEncoder function can be expressed as:

$$\text{TransformerEncoder}(V) = \text{LayerNorm}(V + \text{FFN}(\text{LayerNorm}(V + \text{MultiHead}(V, V, V)))) \tag{5}$$

for each of the L encoder layers (where L is a hyperparameter), with outputs of one layer becoming inputs to the next. The function g is a problem-specific "decoder" pooling + MLP layer that maps the Transformer Encoder's learned salient latent player embeddings to the desired label space $\mathcal{Y}$. The pooling operation is necessary for problems where when we need to eventually aggregate information across all latent player embeddings for a single shared prediction. In Figure 1, we visualize this architecture at a high level, demonstrating its generalizability across problems in Sports Tracking. The key advantage of this architecture lies in its self-attention mechanism. As visualized in the center of Figure 1, each player vector is updated with information from every other player in a weighted, parameterized manner. The model learns to identify which player interactions are important for the prediction task, creating rich embeddings that capture complex spatial relationships. This end-to-end learning of spatial interactions, without requiring manual feature engineering, is fundamental to the approach and applies broadly across sports tracking modeling tasks. This approach addresses the limitations of previous methods in several ways:

1. It operates directly on raw player vectors, eliminating the need for feature engineering or fixed ordering schemes.
2. The permutation equivariance property naturally handles the player-ordering problem.
3. The self-attention mechanism allows for learning complex, long-range spatial relationships between players.
4. The architecture is flexible and can be adapted to various sports and analytical tasks with minimal modifications.

Importantly, this architecture is inherently sport-agnostic and feature-flexible. As illustrated in Figure 1, the same Transformer Player Encoder can process tracking data from basketball, soccer, American football, or any team sport with spatiotemporal player tracking. The model accepts any set of player feature vectors $v_k \in \mathbb{R}^d$, where d can vary based on available features—whether positions, velocities, orientations, player characteristics, or sport-specific attributes. This flexibility allows practitioners to leverage whatever tracking data is available in their domain without architectural modifications. This generalizability is validated by related work successfully applying attention mechanisms to basketball [Alcorn and Nguyen, 2021] and soccer [Wang et al., 2024], supporting our belief that Transformer-based approaches represent a unified solution for multi-agent sports modeling.

[2] While we do not dive into the details of the internal pieces of the Transformer, we note that they have been proven to be highly effective learners in many domains. For a comprehensive understanding of Transformer internals, we recommend many of the excellent public sources on the subject

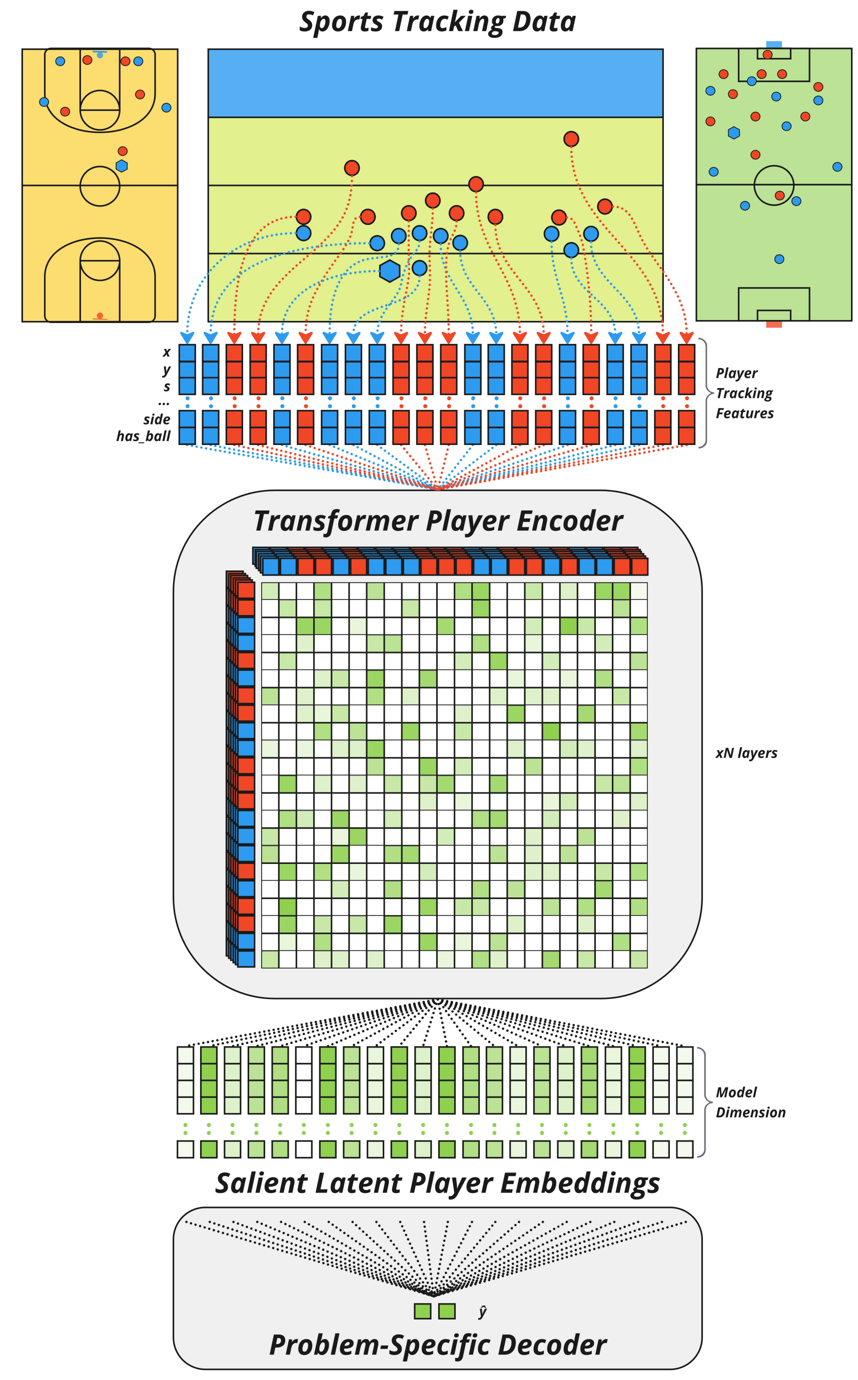

Figure 1: Generalized end-to-end Transformer architecture for sports tracking data. The architecture consists of: (1) an input layer ingesting raw tracking features as unordered player vectors, (2) a Transformer Player Encoder that transforms these into salient player embeddings through L repeated Multi-Head Attention layers, and (3) a problem-specific decoder that pools information (when needed) from the embeddings to produce the final prediction $\hat{y}$. The center of the figure visualizes a single head of self-attention—the key mechanism enabling this approach for multi-agent problems. Each player vector is updated with weighted information from every other player, allowing the model to learn which spatial relationships matter for the prediction task. This attention-based interaction between all players creates rich contextual embeddings, a pattern we believe generalizes broadly across sports tracking modeling tasks.

2.5 Prior Equivariant Solutions

This paper is not the first to develop general-purpose Deep Learning frameworks over unordered player tracking in sports. Here, we compare several notable approaches to our proposed Transformer-based method:

2.5.1 DeepSets

Horton [2020] targeted the problem of proposing a canonical end-to-end modeling of raw trajectory data for learning latent representations generally across problems and sports. This paper, released before Transformers demonstrated widespread modeling success outside of the NLP domain, used an equivariant architecture *DeepSets* [Zaheer et al., 2018] that relies on global pooling of element-wise transformations.

While DeepSets offers permutation equivariance, it is likely inferior to Transformers in sports modeling scenarios where the set size is small but require deep, long-range pattern recognition. Transformers' self-attention mechanism allows for more nuanced interactions between all players, capturing complex spatial and temporal dependencies crucial in team sports. Moreover, Transformers have become ubiquitous across various domains, leading to extensive research, optimizations, and pre-trained models, which DeepSets lack.

2.5.2 Graph Neural Nets

Yeh et al. [2019] proposed using graph neural networks (GNNs) as a permutation-equivariant method to model over unordered players where each player represents a node in a fully connected graph.

While GNNs offer a natural representation of players on a field, Transformers present several advantages in the context of sports modeling with fully connected player interactions:

- In a fully connected scenario, the multi-hop message passing of GNNs becomes redundant, as all nodes are directly connected. Transformers, with their self-attention mechanism, can model these direct interactions more efficiently.
- The sparsity benefits typically associated with GNNs are nullified in a fully connected graph, negating one of their key advantages.
- Alcorn and Nguyen [2021] demonstrates empirically that Transformers outperform this specific GNN approach.

2.5.3 The Zoo

The Zoo [2020] is a research group developed a model architecture that won the 2020 NFL Big Data Bowl in Kaggle challenge. This victory, along with the Big Data Bowl's growing prominence, led to The Zoo Architecture (hereafter referred to as Zoo) becoming a de-facto equivariant deep learning approach used by entrants in following competitions. Furthermore, this architecture powers models and stats distributed by the NFL's advanced wing *Next Gen Stats*. In their submission, they identified the importance of designing for the player-order equivariance problem but discovered that their custom equivariant Zoo design *outperformed* Transformers with Multi-Head Attention. Zoo, as shown in Figure 2, relies on feature engineering vectors pairwise between each offensive and defensive player, and then operating dense layers over each pairwise vector *independently*, with pooling operations to eventually reduce the dimensionality into one final prediction. While not explicitly cited, we find many similarities between Zoo and the DeepSet approach. We expect Zoo to be architecturally inferior for similar reasons:

- It is not a true end-to-end approach, relying on manual feature engineering.
- The independent processing of pairwise vectors may limit the model's ability to capture complex, multi-player interactions.
- The reliance on multiple intermediate pooling operations may lead to loss of important spatial information.

Furthermore, in our Experiments (Section 3), we demonstrate this inferiority empirically in a similar, but slightly more general task than what Zoo was originally optimized for. We find that while Zoo may have achieved impressive performance for that original dataset and specific task, it does not extend generally to other problems, even those that are just slightly different.

3 Experiments

The model code, data, and results for our experiments is available here.

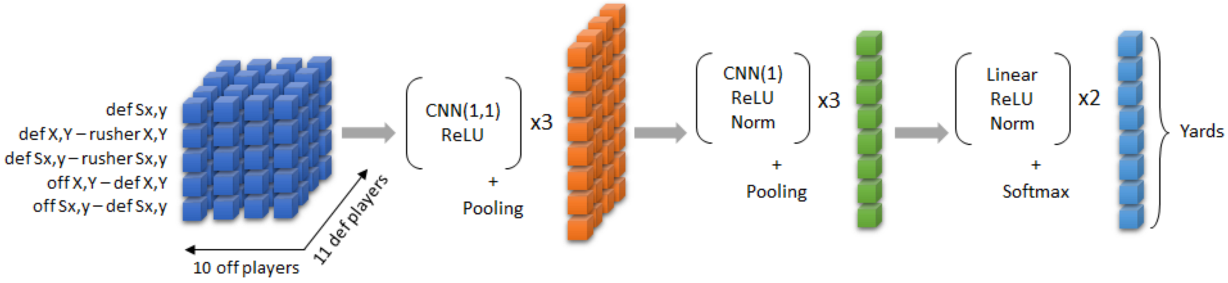

Figure 2: Simplified Structure of The Zoo Architecture that rose to prominence in modeling spatiotemporal NFL tracking data. It was designed to predict a categorical distribution over the number of yards gained on rushing plays using tracking frames at time of handoff. The model relies on manually constructing "interaction" feature vectors pairwise between each offensive and defensive player. The model then essentially treats these "interaction" feature vectors as independent throughout, with no mechanism to learn across player dimension. After applying a few dense layers to each interaction vector, pooling is applied to collect the most salient learned features across the offensive player and then defensive player dimension.

3.1 Dataset

We utilized data from the 2024 NFL Big Data Bowl, a public competition hosted on Kaggle[3]. This dataset provides comprehensive tracking data, including the location, speed, and orientation of all 22 players on the field for Weeks 1-9 of the 2022 NFL Season. The dataset's focus on tackling aligns well with our research goals, as it presents a complex spatial task that requires understanding player interactions. Key characteristics of the dataset include:

- Coverage: 136 games, approximately 2,000 unique plays, and 80,000 frames
- Content: Tracking frames where there is a clear ball-carrier and the defense is focused on tackling
- Features: Player positions, velocities, and orientations for each frame

For our modeling objective, we chose to predict the (x, y) position of the tackle. This task was selected for several reasons:

1. Alignment with the dataset's tackling theme
2. Similarity to tasks that previous equivariant architectures (e.g., Zoo) were designed for, allowing for meaningful comparisons
3. Generalizability across various football situations (e.g., post-handoff, post-pass catch, during scrambles)
4. Continuous output space, presenting a regression problem rather than a classification task

To standardize the data, we followed common practices in NFL modeling:

- All plays were standardized so that the offense always moves to the right
- We mirrored the data across the y-axis, effectively doubling our training dataset
- Normalize x, y positions to be relative to an anchor defined as the location of the ball-carrier at the start of a play. This is done primarily to ensure tracking frames look to be drawn from a similar distribution rather than different values based on the yardline of the play, and not for ordering the players in any way.
- We excluded the ball location data as in this dataset the ball was always held by a ball carrier and so was redundant.

For our experiments, we treated each frame as an independent input, although the tackle location labels are unique at the play level. We split the data into training, validation, and test sets with a 70/15/15 ratio, randomly sampling at the play level (seed=42). This resulted in:

- Training set: 9,000 unique plays, 750,000 frames
- Validation set: 2,000 unique plays, 150,000 frames
- Test set: 2,000 unique plays, 150,000 frames

[3] https://www.kaggle.com/competitions/nfl-big-data-bowl-2024/data

3.2 Evaluation Metric

We evaluate model performance using Average Displacement Error (ADE), a standard metric for spatial prediction tasks. ADE measures the average Euclidean distance between predicted and actual tackle locations:

$$\text{ADE} = \frac{1}{N}\sum_{i=1}^{N}\sqrt{(x_{\text{pred}}^{(i)} - x_{\text{true}}^{(i)})^2 + (y_{\text{pred}}^{(i)} - y_{\text{true}}^{(i)})^2} \tag{6}$$

where N is the number of predictions, and coordinates are measured in yards on the football field. Lower ADE values indicate more accurate tackle location predictions.

3.3 Models

To evaluate the effectiveness of the Transformer architecture in sports analytics, we compare it against The Zoo Architecture, a baseline model that has shown success in similar tasks. To ensure a fair comparison, both models share the same AdamW optimizer, learning rate, and SmoothL1Loss function (also known as Huber Loss). The SmoothL1Loss balances L1 Loss for outliers with L2 Loss for predictions close to the target, providing robustness and numerical stability to our training. We conducted a grid search over learning rates, model sizes, and number of layers for both models, while keeping batch size and dropout fixed. This approach allows us to isolate the impact of architectural differences on model performance. The results are from the best model picked over the grid search. Models trained for maximum 200 epochs with early stopping (patience=10).

3.3.1 Transformer Model Experiment

For the Transformer model in this experiment, we define the set of feature vectors $V = \{v_1, v_2, ..., v_K\}$ over the $K = 22$ players, where each $v_k \in \mathbb{R}^6$ represents the features of player p_k. Specifically, each feature vector v_k is composed of:

$$v_k = [x_k, y_k, vx_k, vy_k, o_k, b_k] \tag{7}$$

where:

- (x_k, y_k) represent the spatial coordinates of player p_k
- (vx_k, vy_k) represent the velocity components of player p_k
- $o_k \in \{0, 1\}$ is a binary indicator for offense (1) or defense (0)
- $b_k \in \{0, 1\}$ is a binary indicator for whether player p_k is the ball carrier (1) or not (0)

The label space $\mathcal{Y}$ for our task is the predicted tackle location, defined as $\mathcal{Y} = \mathbb{R}^2$, representing the (x, y) coordinates of the predicted tackle.

The TransformerEncoder model $f : (\mathbb{R}^6)^{22} \to (\mathbb{R}^d)^{22}$ maps the set of player feature vectors V equivariantly into a set of player embeddings of size model dimension d. Then we apply the task-specific decoder $g : (\mathbb{R}^d)^{22} \to \mathbb{R}^2$ as an average pooling over the players followed by an MLP to get the predicted tackle location:

$$\hat{y} = (\hat{x}_{tackle}, \hat{y}_{tackle}) = g(f(V)) = g(f(\{v_1, v_2, ..., v_{22}\})) \tag{8}$$

Implementation: $d_{\text{ff}} = 4d$, attention heads $= \min(16, \max(2, 2 \times \lfloor d/64 \rfloor))$, BatchNorm1d for normalization, AdaptiveAvgPool1d for pooling. In our code implementation, L corresponds to the `num_layers` hyperparameter.

3.3.2 The Zoo Architecture Model Details

For our Zoo, we have to perform a complex pairwise feature interaction process $\Phi : V \to (\mathbb{R}^{10})^{10 \cdot 11}$ that converts $V = \{v_1, v_2, ..., v_K\}$ over the $K = 22$ players into 110 interaction vectors. Specifically, each interaction vector u_{ij} between offensive player p_i and defensive player p_j with ball carrier p_b is composed of:

$$u_{ij} = [vx_j, vy_j, x_j - x_b, y_j - y_b, vx_j - vx_b, vy_j - vy_b, x_i - x_j, y_i - y_j, vx_i - vx_j, vy_i - vy_j] \tag{9}$$

where:

- (x_k, y_k) represent the spatial coordinates of player p_k
- (vx_k, vy_k) represent the velocity components of player p_k

- p_b is the ball carrier

Then Zoo model $g : (\mathbb{R}^{10})^{10 \cdot 11} \rightarrow \mathbb{R}^2$ applies successive MLPs to interaction vectors independently, only pooling across to reduce dimensionality.

Contrasting Architectural Philosophies: The key distinction between these architectures lies in their approach to learning player interactions. Zoo requires explicit, hand-crafted feature engineering through the pairwise interaction vectors defined in Equation (8): a manual design choice that encodes domain assumptions about which interactions matter. In contrast, the Transformer architecture learns these interactions end-to-end via the attention mechanism, discovering relevant pairwise relationships directly from the raw player features without requiring manual specification of interaction patterns. This fundamental difference exemplifies the shift from feature engineering to end-to-end learning in deep learning research.

4 Results

4.1 Model Selection and Architectural Scaling

Before comparing final model performance, we examine how each architecture responds to increased model capacity. This analysis validates that our findings are not simply due to selecting a larger Transformer model, and reveals fundamental differences in architectural expressiveness.

We trained 24 models total (12 per architecture) spanning the following hyperparameter space: model dimensions {32, 128, 512}, number of layers {1, 2, 4, 8}, with fixed learning rate (1e-4), batch size (256), and dropout (0.3). Table 2 presents representative configurations from both architectures.

Table 2: Representative Model Configurations Showing Scaling Behavior

Model	**Dim**	**Layers**	**Params**	**FLOPs**	**Test ADE (yards)**
Zoo Architecture					
Zoo-Small	32	2	5K	0.6M	6.25
Zoo-Medium	128	2	73K	8.4M	**5.78**
Zoo-Large	512	2	1.1M	128.9M	5.77
Zoo-XLarge	512	8	5.9M	513.5M	5.89
Transformer Architecture					
Transformer-Small	32	4	52K	2.2M	4.95
Transformer-Medium	128	2	418K	17.5M	4.60
Transformer-Large	512	2	6.6M	277.9M	**4.61**
Transformer-XLarge	512	8	25.6M	1109.1M	4.65

The results reveal strikingly different scaling behaviors between the two architectures. The optimal Zoo configuration is the relatively modest Zoo-Medium (73K parameters, 8.4M FLOPs, 5.78 yards ADE). Increasing capacity beyond this point consistently degrades performance: Zoo-XLarge, despite being 81× larger in parameters and 61× larger in FLOPs, achieves worse performance at 5.89 yards. Larger Zoo models also reached early stopping substantially earlier (Zoo-Medium at epoch 49 vs. Zoo-Large at epoch 8 vs. Zoo-XLarge at epoch 3), a pattern consistent with rapid overfitting that prevents these models from effectively utilizing additional capacity. This pattern held across our entire hyperparameter sweep, with larger Zoo models universally underperforming their smaller counterparts.

In contrast, the Transformer shows clear improvements with increased capacity: 4.95 → 4.60 → 4.61 yards as we scale from Small to Medium to Large configurations (127× parameter increase). The best-performing Transformer-Large model trained for 12 epochs before early stopping, demonstrating that it could effectively leverage its increased capacity to achieve superior performance. Only at extreme capacity (Transformer-XLarge at 25.6M parameters, 6 epochs) do we observe slight overfitting.

Critically, Transformer models outperform Zoo even at matched computational budgets. Transformer-Small (52K params, 2.2M FLOPs) achieves 4.95 yards, substantially better than Zoo-Medium (73K params, 8.4M FLOPs) at 5.78 yards. This represents a 14% improvement with fewer parameters and less computation. This pattern holds across all comparable model sizes, with Transformers consistently achieving 15-23% better performance at matched FLOP budgets.

These results demonstrate that the Transformer's advantages stem from fundamental architectural properties, specifically its ability to capture higher-order interactions between all players through self-attention, rather than simply having more parameters.

Figure 3 visualizes these scaling behaviors, showing how Zoo architecture peaks early and degrades with increased capacity, while Transformer architecture continues to improve with scaling and outperforms Zoo at matched computational budgets.

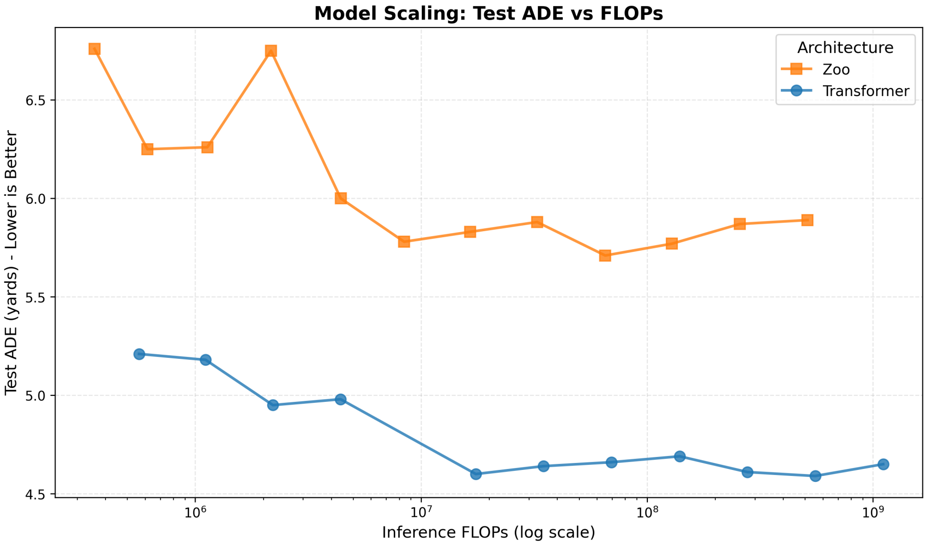

Figure 3: Model Scaling: Test ADE vs FLOPs. Zoo architecture (orange) peaks at Medium (73K params, 8.4M FLOPs) then degrades with more capacity (5.78 → 5.89 yards), while Transformer architecture (blue) improves with scaling (4.95 → 4.60 → 4.61 yards). The Transformer consistently outperforms Zoo at matched computational budgets (15-23% better), demonstrating that its advantages stem from fundamental architectural properties rather than simply having more parameters.

For the detailed performance comparisons that follow, we selected the best-performing model from each architecture irrespective of size: Zoo-Medium (73K params, 5.78 yards) and Transformer-Large (6.6M params, 4.61 yards). While these models differ in capacity by 91× in parameters and 33× in FLOPs, this comparison reflects the optimal version of each architectural approach and deliberately avoids penalizing the Transformer for Zoo's inability to scale.

4.2 Overall Performance

Our evaluation on a held-out test set reveals that the Transformer model substantially outperforms Zoo by 20.2% in Average Displacement Error (ADE), measured in yards. Table 3 shows performance across all data splits.

Table 3: Overall Performance Comparison Across Data Splits

Split	Plays	Zoo (yards)	Transformer (yards)	Improvement %	Improvement (yards)
Train	8,735	5.01	4.03	19.6%	0.98
Val	1,872	5.81	4.69	19.3%	1.12
Test	1,871	5.78	4.61	20.2%	1.17

4.3 Performance by Event Type

A breakdown of performance by specific game events provides further insights into the strengths of each model (Table 4).

Notably, both models perform comparably for early-play events such as ball snap and handoff. Zoo's strong performance in these situations aligns with its original optimization for handoff-related tasks, demonstrating its effectiveness in specific, well-defined scenarios.

Table 4: Test Set Event-Frame Performance Comparison (Average Displacement Error in Yards)

Event	Plays	Zoo (yards)	Transformer (yards)	Improvement %	Improvement (yards)
Ball Snap	958	8.72	8.77	-0.6%	-0.05
Handoff	887	6.69	6.59	1.5%	0.10
Run	142	7.92	6.83	13.8%	1.09
Pass Arrived	732	5.13	4.72	8.0%	0.41
Pass Outcome Caught	842	4.62	4.22	8.7%	0.40
First Contact	1,578	4.06	2.88	29.1%	1.18
Out of Bounds	272	5.81	1.72	70.4%	4.09
Tackle	1,497	4.07	0.98	75.9%	3.09

However, the Transformer model's advantage becomes increasingly apparent in later-play events, particularly out of bounds and tackle situations. In these cases, where the prediction target is essentially the ball carrier's current location, the Transformer shows a superior ability to recognize play-ending situations and adapt to diverse game contexts.

This pattern is continued when we analyze performance based on temporal distance from the frame of tackle (Figure 4).

4.4 Performance by Frames Before Tackle

Table 5 presents a detailed breakdown of model performance at different temporal distances from the tackle. This analysis reveals how the Transformer's architectural advantages manifest across the progression of each play.

Table 5: Test Set Performance by Frames Before Tackle

Frames Before Tackle	Frames	Zoo (yards)	Transformer (yards)	Improvement %	Improvement (yards)
After tackle	7,480	4.61	1.16	74.8%	3.45
0-5 frames	9,355	4.27	1.22	71.4%	3.05
5-10 frames	9,248	3.88	1.52	60.8%	2.36
10-15 frames	8,679	3.61	2.23	38.2%	1.38
15-20 frames	7,952	3.78	3.13	17.2%	0.65
20-25 frames	7,228	4.34	4.19	3.5%	0.15
25-30 frames	6,443	5.12	5.21	-1.8%	-0.09
30+ frames	23,677	9.57	9.57	0.0%	0.00

The data reveals a striking contrast in how the two architectures handle predictions as plays develop. Far from the tackle (30+ frames), both models perform similarly poorly (9.5 yards error) when the play outcome is highly uncertain. However, as plays progress toward the tackle, the models diverge dramatically:

The Transformer continuously improves, leveraging increasing spatial information: 9.57 → 5.21 → 4.19 → 3.13 → 2.23 → 1.52 → 1.22 → 1.16 yards error.

The Zoo model actually degrades near critical moments: after reaching its best performance at 10-15 frames (3.61 yards), it gets progressively worse as the tackle approaches: 3.61 → 3.88 (5-10 frames) → 4.27 (0-5 frames) → 4.61 (after tackle).

This performance degradation near the tackle (where error should be minimal as we approach the play's conclusion) suggests that the Zoo architecture struggles to adapt to the dynamic spatial patterns that emerge as plays evolve beyond their initial configurations. In contrast, the Transformer's self-attention mechanism demonstrates robust generalization across diverse game situations, achieving 60.8% to 74.8% improvement near the tackle. Figure 4 visualizes this pattern.

Figure 5 shows a visual example of the Transformer generalizing to unseen frames in the test set significantly better than Zoo is able to.

5 Supporting Work

Our discussion thus far has focused on previous work that relied on decomposing the modeling process through feature engineering (Section 2.3) and approaches that proposed solutions for handling player-order equivariance without

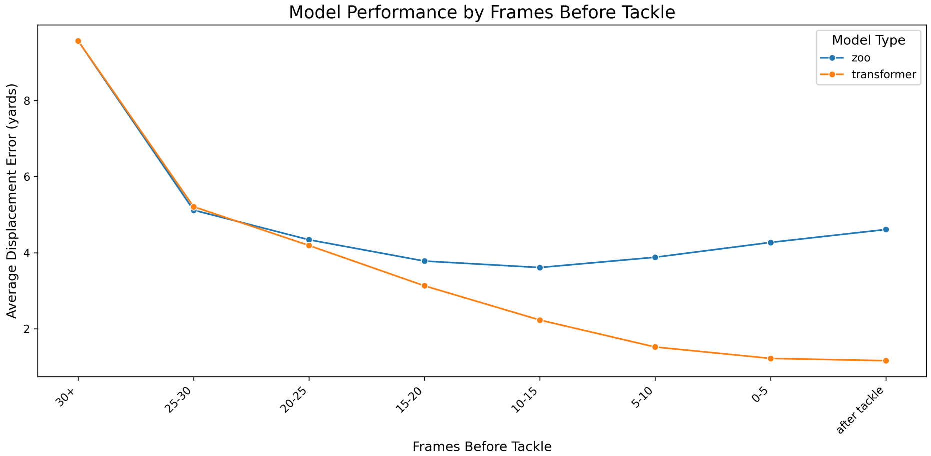

Figure 4: Performance by Frames Before Tackle. The plot reveals a critical architectural difference: while the Transformer (blue) continuously improves as plays progress toward the tackle (9.57 → 1.16 yards), the Zoo model (orange) actually degrades near critical moments (3.61 → 4.61 yards from 10-15 frames to after tackle). Both models perform similarly far from the tackle (30+ frames, 9.6 yards), but the Transformer's self-attention mechanism excels at leveraging rich spatial information during crucial moments, achieving up to 74.8% improvement after the tackle.

utilizing Transformers or Attention mechanisms (Section 2.5). To provide a comprehensive overview, it is crucial to highlight research that has employed Attention mechanisms for addressing similar challenges in sports analytics:

- **Baller2Vec:** Alcorn and Nguyen [2021] proposed an innovative approach for analyzing basketball trajectories. Their method employed masked Attention applied causally across the time dimension while simultaneously allowing for equivariant learning across the player dimension. While we strongly endorse this work, we posit that its broader scope may have inadvertently obscured the value of Attention mechanisms for static, single-frame use cases, potentially limiting its widespread adoption in sports analytics.

- **TacticAI:** Wang et al. [2024] addressed a problem similar to the one presented in this paper, focusing on predicting events for corner kicks from static tracking frames. Their approach represents players as nodes in a graph, akin to the method discussed in Section 2.5.2, but employs the more advanced Graph Attention v2 (GATv2) mechanism for learning over the graph structure. While GATv2 is a sophisticated modification of the original Attention mechanism tailored for canonical graph representation problems, we argue that its application may be excessive in the context of sports tracking data. In this domain, the graph is fully connected, contains relatively few nodes, and lacks significant edge features. Consequently, we posit that the application of GATv2 principles in this specific scenario likely reduces to a formulation very similar to the simpler Transformer Attention mechanism proposed in our work.

These studies underscore the utility of Attention mechanisms for modeling sports tracking frames. However, we believe that our paper contributes significantly by providing an easily reproducible approach with a narrower scope, focusing specifically on the application of Transformer-based Attention to static, single-frame sports tracking data.

5.1 Limitations and Future Directions

This work focuses deliberately on a narrow empirical scope to provide a clear, reproducible demonstration of Transformer effectiveness. While our architecture is designed to be both sport-agnostic and feature-flexible—accepting any set of spatiotemporal player features regardless of sport or tracking system—we evaluated it on a single task (tackle prediction) in one sport (NFL) using static, single-frame inputs without temporal modeling across frames. The architectural principles and the success of related attention-based approaches in basketball and soccer strongly suggest broad applicability, but empirical validation across diverse sports, tasks, and temporal modeling approaches remains valuable future work. We welcome further research to explore these dimensions and extend our findings.

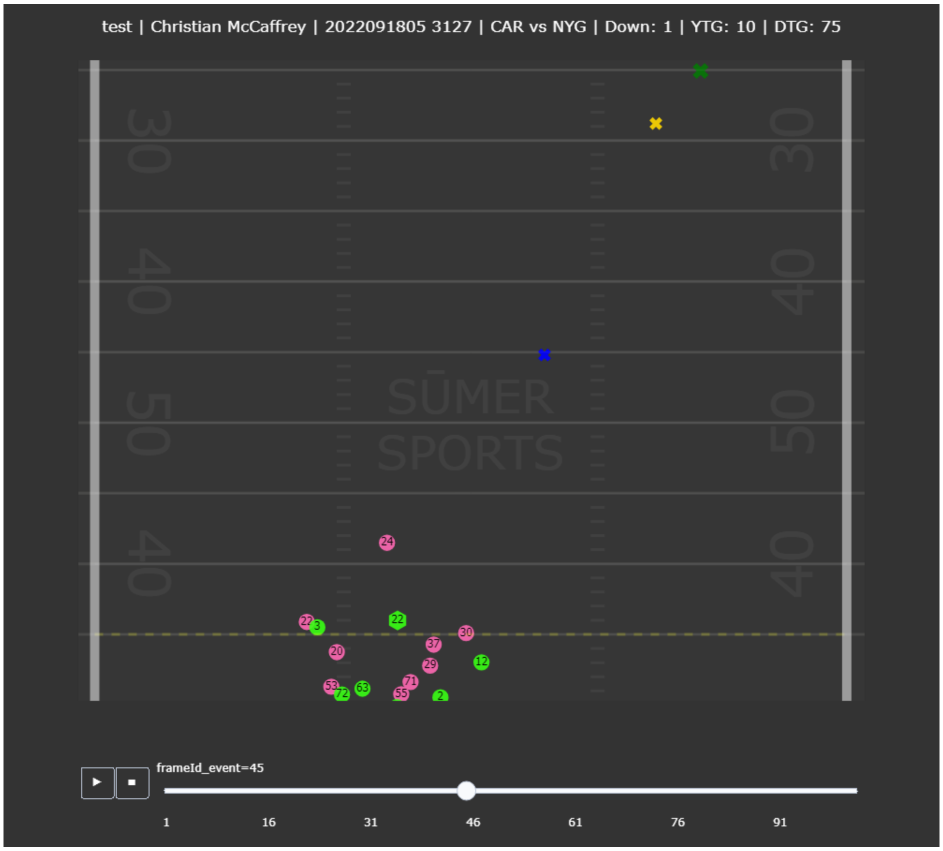
test | Christian McCaffrey | 2022091805 3127 | CAR vs NYG | Down: 1 | YTG: 10 | DTG: 75
30
40
50
40
30
40
50
40
SŪMER
SPORTS
24
27
3
22
30
20
37
12
29
53
72
63
71
55
2
▶
■
frameId_event=45
1
16
31
46
61
76
91

Figure 5: Visualization of a frame from the test set showing improved generalization from the Transformer. The green hexagon (#22) is the ball-carrier, green cross is the true tackle location. Yellow and blue crosses represent predictions from Transformer and Zoo models respectively

6 Conclusion

In summary, the adoption of transformers in sports data modeling promises to address the limitations of existing methodologies by providing a simple, generalized, and scalable framework. Our methodology demonstrates the superior performance of transformer-based models compared to traditional approaches like The Zoo Architecture, highlighting their potential in capturing complex spatial interactions with minimal feature engineering. This paradigm shift could facilitate more robust and flexible analyses, ultimately advancing the field of sports data science.

6.1 Further Work

It was with intention that we kept the experiments and scope of this paper narrow. We wanted to stoke interest in Transformers as a powerful tool to be used in sports modeling, but leave plenty of room for innovation and further work to extend this. For example, while we have made claims about the generalizability of this approach we only had the resources to explore one application in this paper in one sport. We welcome additional effort to rigorously compare it to solutions that are not end-to-end and in other data spaces.

References

Stephanie A. Kovalchik. Player tracking data in sports. *Annual Review of Statistics and Its Application*, 10(Volume 10, 2023):677–697, 2023. ISSN 2326-831X. doi:https://doi.org/10.1146/annurev-statistics-033021-110117. URL `https://www.annualreviews.org/content/journals/10.1146/annurev-statistics-033021-110117`.

Michael Horton. Learning feature representations from football tracking. In *MIT Sloan Sports Analytics Conference*, Boston, March 2020. MIT sloan sports analytics conference

Boston, MA, USA. URL https://www.sloansportsconference.com/research-papers/learning-feature-representations-from-football-tracking. Presented on March 6–7, 2020.

Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015. doi:10.1038/nature14539.

Andrew Ng. *Machine Learning Yearning*. deeplearning.ai, Mountain View, CA, 2018. URL https://info.deeplearning.ai/machine-learning-yearning-book. Section 47: "The rise of end-to-end learning", pages 91–96.

Javier Fernández, Luke Bornn, and Dan Cervone. Decomposing the immeasurable sport: A deep learning expected possession value framework for soccer. In *13th MIT Sloan Sports Analytics Conference*, volume 2, 2019.

Ronald Yurko, Francesca Matano, Lee F Richardson, Nicholas Granered, Taylor Pospisil, Konstantinos Pelechrinis, and Samuel L Ventura. Going deep: models for continuous-time within-play valuation of game outcomes in american football with tracking data. *Journal of Quantitative Analysis in Sports*, 16(2):163–182, 2020.

Anar Amirli and Hande Alemdar. Prediction of the ball location on the 2d plane in football using optical tracking data. *Academic Platform Journal of Engineering and Smart Systems*, 10(1):1–8, 2022.

Hoang M Le, Peter Carr, Yisong Yue, and Patrick Lucey. Data-driven ghosting using deep imitation learning. In *MIT Sloan Sports Analytics Conference*. MIT sloan sports analytics conference Boston, MA, USA, 2017.

Marc Schmid, Patrick Blauberger, and Martin Lames. Simulating defensive trajectories in american football for predicting league average defensive movements. *Frontiers in Sports and Active Living*, 3, 2021. ISSN 2624-9367. doi:10.3389/fspor.2021.669845. URL https://www.frontiersin.org/journals/sports-and-active-living/articles/10.3389/fspor.2021.669845.

Panna Felsen, Patrick Lucey, and Sujoy Ganguly. Where will they go? predicting fine-grained adversarial multi-agent motion using conditional variational autoencoders. In *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

Nazanin Mehrasa, Yatao Zhong, Frederick Tung, Luke Bornn, and Greg Mori. Deep learning of player trajectory representations for team activity analysis. In *11th mit sloan sports analytics conference*, volume 2, page 3, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017. URL https://arxiv.org/abs/1706.03762.

Michael A. Alcorn and Anh Nguyen. baller2vec: A multi-entity transformer for multi-agent spatiotemporal modeling, 2021. URL https://arxiv.org/abs/2102.03291.

Zhe Wang, Petar Veličković, Daniel Hennes, Nenad Tomašev, Laurel Prince, Michael Kaisers, Yoram Bachrach, Romuald Elie, Li Kevin Wenliang, Federico Piccinini, William Spearman, Ian Graham, Jerome Connor, Yi Yang, Adrià Recasens, Mina Khan, Nathalie Beauguerlange, Pablo Sprechmann, Pol Moreno, Nicolas Heess, Michael Bowling, Demis Hassabis, and Karl Tuyls. Tacticai: an ai assistant for football tactics. *Nature Communications*, 15(1):1906, 3 2024. ISSN 2041-1723. doi:10.1038/s41467-024-45965-x. URL https://doi.org/10.1038/s41467-024-45965-x.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets, 2018. URL https://arxiv.org/abs/1703.06114.

Raymond A. Yeh, Alexander G. Schwing, Jonathan Huang, and Kevin Murphy. Diverse generation for multi-agent sports games. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4605–4614, 2019. doi:10.1109/CVPR.2019.00474.

The Zoo. 1st place solution - nfl big data bowl 2020. Kaggle, 2020. URL https://www.kaggle.com/c/nfl-big-data-bowl-2020/discussion/119400. Winning submission for the NFL Big Data Bowl 2020.