



# Transforming Deep Learning in Football

Udit Ranasaria, Vishakh Sandwar, Pavel Vabishchevich

*CMSAC Football Analytics Workshop, Oct 2025*



A Typical Saquon Play

 SUMERSPORTS

# Previous Deep Learning in Football

## Tracking Data Everywhere

- 2016: Sensor 2D Tracking
  - RFID chips in pads
  - Zebra + NGS
- 2020: CV 2D Tracking
  - Generated from All-22 Film
  - SportLogiq / Telemetry
  - Many others now!
- Optical 3D Tracking
  - Pose estimation
  - Exists in other sports
  - Hawk-Eye recently deployed in NFL 



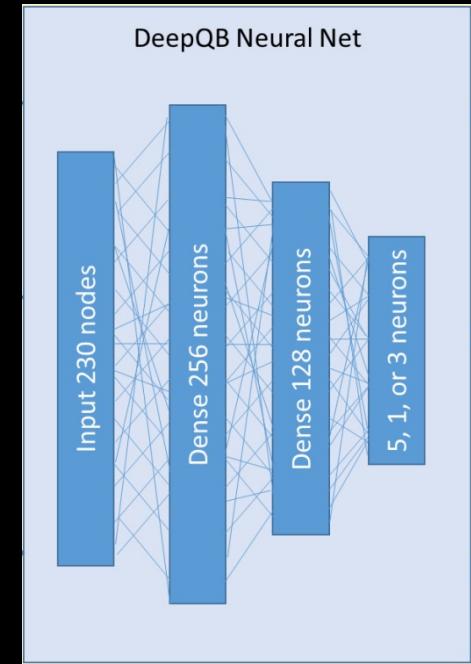
## 2D Tracking Data Form

- Each frame represents a moment in time
- Spatial features for each player
- Bonus: Events, ball info, etc.
- For today, scope to modeling assuming each frame represents an independent data sample

FramedID	Event	Player ID	Team	x	y	Has Ball
15	Handoff	80932	OFF	26.9	27.9	0
15	Handoff	87432	OFF	35.2	34.0	1
15	Handoff	09323	OFF	40.4	25.4	0
:	:	:	:	:	:	:
15	Handoff	78152	DEF	35.3	31.4	0

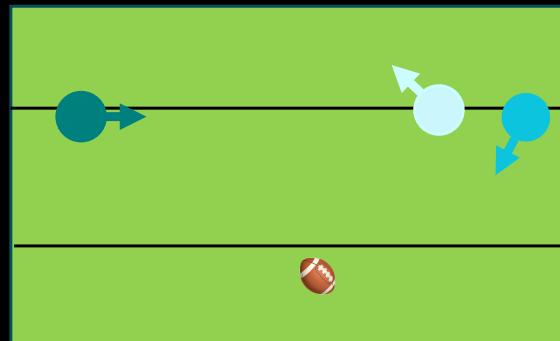
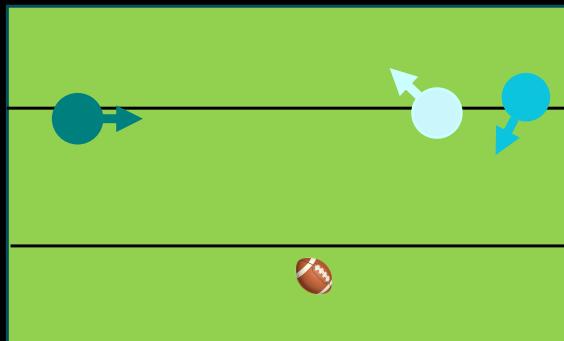
## Getting Started: Fully-Connected Feedforward Networks (FFN)

- Also known as Multilayer Perceptron (MLP)
- The original NN designed for tabular data
- Alternative to other black-box tabular approaches like XGBoost
- Notable Work:
  - DeepQB (Burke 2019)
  - Going Deep (Yurko et al. 2019)
- Limitations:
  - Have to collapse frame into wide-form table of features
  - Forces a heuristic for ordering players
  - Similar tracking data can look very different to model



Burke 2019

## Illustrating the Player Ordering Problem



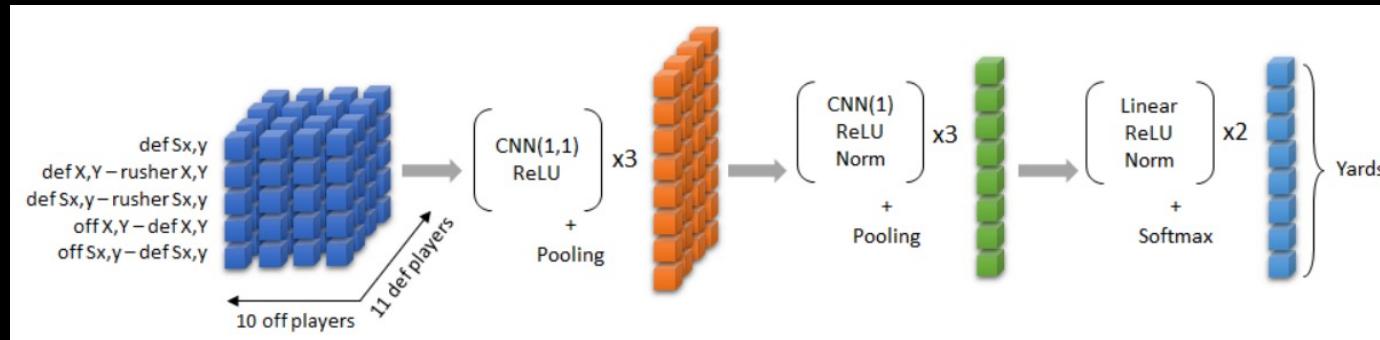
4.3 yards



5.1 yards

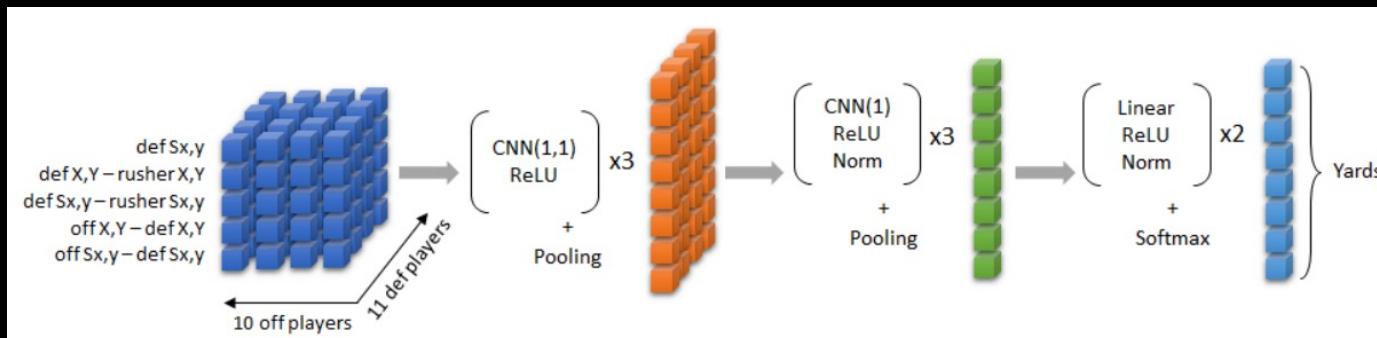


## The Zoo Architecture, a Big Leap Forward



- Singer and Gordeev's legendary win in 2020 BDB
- Identified importance of player-order invariance
- Relied on pooling as an operation that was invariant to player order
  - Pooling is a fancy term for “take the mean or max over” a set of vectors

## The Zoo Architecture: Convolutions??



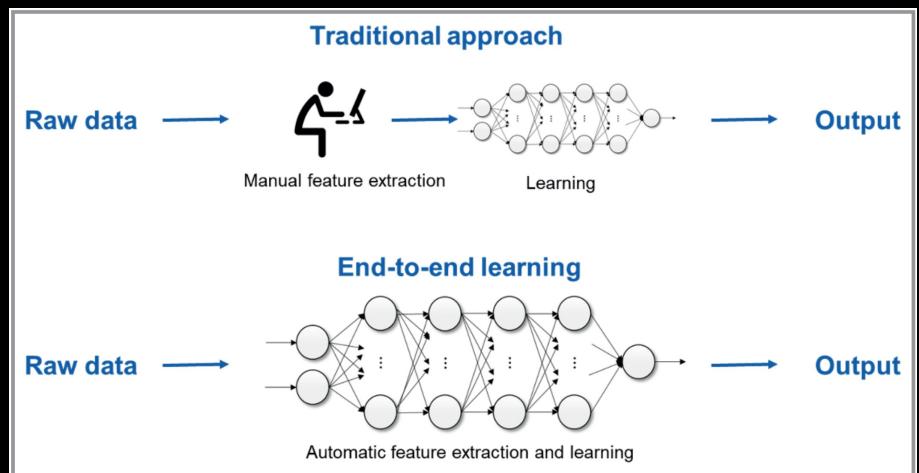
- Generated a cube of interaction vectors between pairwise players
- Applied a  $1 \times 1$  “convolution” over each interaction vector independently
  - $1 \times 1$  convolution is algebraically identical to FFN applied across just one dimension
  - True convolutions would have violated player-order invariance!
- Pool across interaction vectors afterwards
- Limitations:
  - Focused learning to only between 2 opposing players at a time
  - Pooling collapses player dimensions
  - Features are highly specific to rushing targets

# The Transformer Revolution



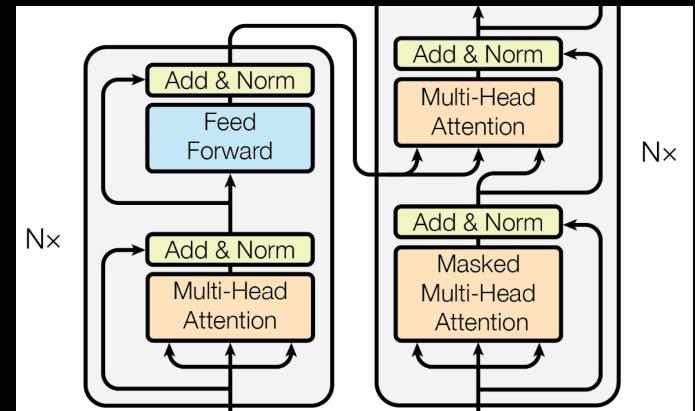
## End-to-End Learning

- Early 2000's: Reliance on hand-crafted features
- Data Volume
- Compute Cost
- Architecture Innovation
- Mid 2010s: Rise of end-to-end learning (AlexNet!)
  - Adapt your architecture to your raw data inputs and outputs.
  - Model will learn salient features in the network given enough data
- For most domains, matter of when not if



## Attention is All You Need

- Seminal Google paper published in June 2017
- Introduced the “Transformer”
  - The “T” in ChatGPT
  - Major reason for the LLM explosion in 2022
  - Key Innovation: Heavy use of “Self Attention”
  - The most used architecture in most AI domains
- Cited 200k+ times (and counting)
- Title inspired by “All you need is Love” by the Beatles
  - Shout out to Claude for this fun fact



# Why Were Transformers So Successful?

- Most data can be seen as a sequence of items
  - Each item has a corresponding vector of data

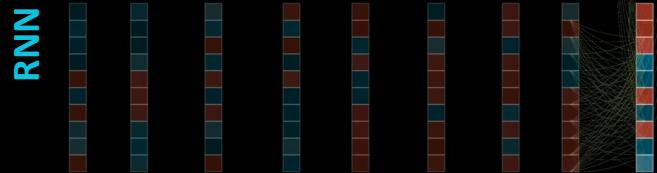
CNNs & RNNs: Order as a heuristic

- Designed to leverage relevant locality, order and adjacency
- Signal between items far apart can get muffled

Transformers: Unordered Bag of Items

- Self-Attention: Direct connections between all pairwise items equally, irrespective of order
- If needed, ordering features can be passed into each item's vector
- Model can learn importance of locality
  - End-to-end learning!
- Bonus: Much more parallel in computation which enabled model scaling for huge LLMs

It was the best of times it was the → ?



It was the best of times it was the → ?



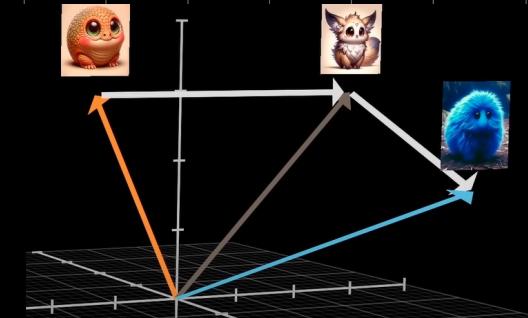
Credit: 3Blue1Brown

## Self Attention in Language

- Pairwise Matrix
- Use item vectors to learn attention weights between pairwise items
- Apply a weighted sum to update each items current vector
- Get an updated output vector for each item after it has been “attended” to by all other items

	a	fluffy	blue	creature	roamed	the	verdant	forest
	$\vec{E}_1$	$\vec{E}_2$	$\vec{E}_3$	$\vec{E}_4$	$\vec{E}_5$	$\vec{E}_6$	$\vec{E}_7$	$\vec{E}_8$
	$W_Q$							
a → $\vec{E}_1 \xrightarrow{W_k} \vec{K}_1$	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
fluffy → $\vec{E}_2 \xrightarrow{W_k} \vec{K}_2$	0.00	1.00	0.00	0.42	0.00	0.00	0.00	0.00
blue → $\vec{E}_3 \xrightarrow{W_k} \vec{K}_3$	0.00	0.00	1.00	0.58	0.00	0.00	0.00	0.00
creature → $\vec{E}_4 \xrightarrow{W_k} \vec{K}_4$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
roamed → $\vec{E}_5 \xrightarrow{W_k} \vec{K}_5$	0.00	0.00	0.00	0.00	0.00	0.01	0.00	0.00
the → $\vec{E}_6 \xrightarrow{W_k} \vec{K}_6$	0.00	0.00	0.00	0.00	0.99	1.00	0.00	0.00
verdant → $\vec{E}_7 \xrightarrow{W_k} \vec{K}_7$	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00
forest → $\vec{E}_8 \xrightarrow{W_k} \vec{K}_8$	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Credit: 3Blue1Brown

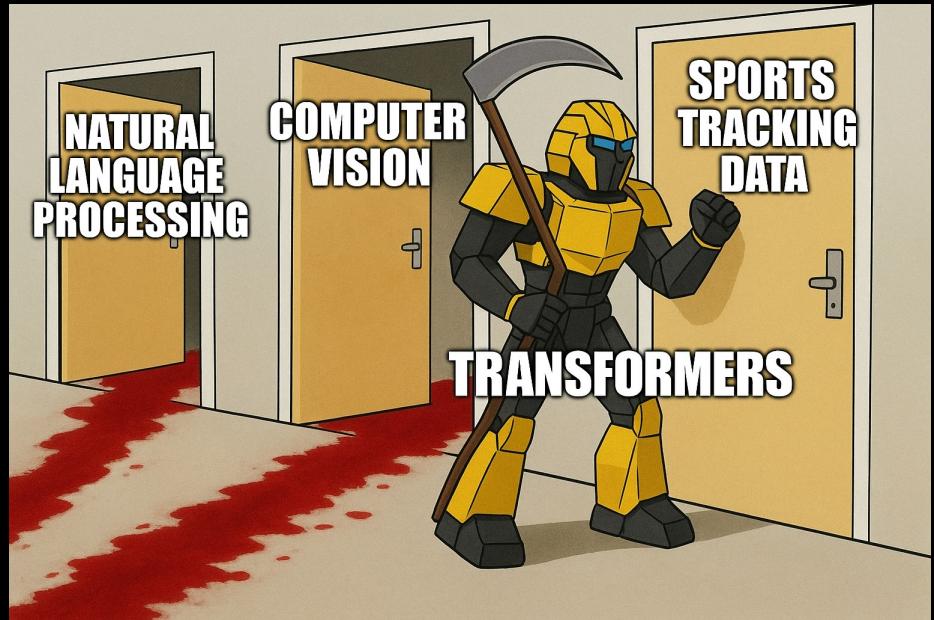


# Transforming Deep Learning in Football



## Recap of Goals

- We want a solution that fits:
  - Player Order Invariance
  - End-to-End Learning
  - General across sports problems
  - Learns well across players

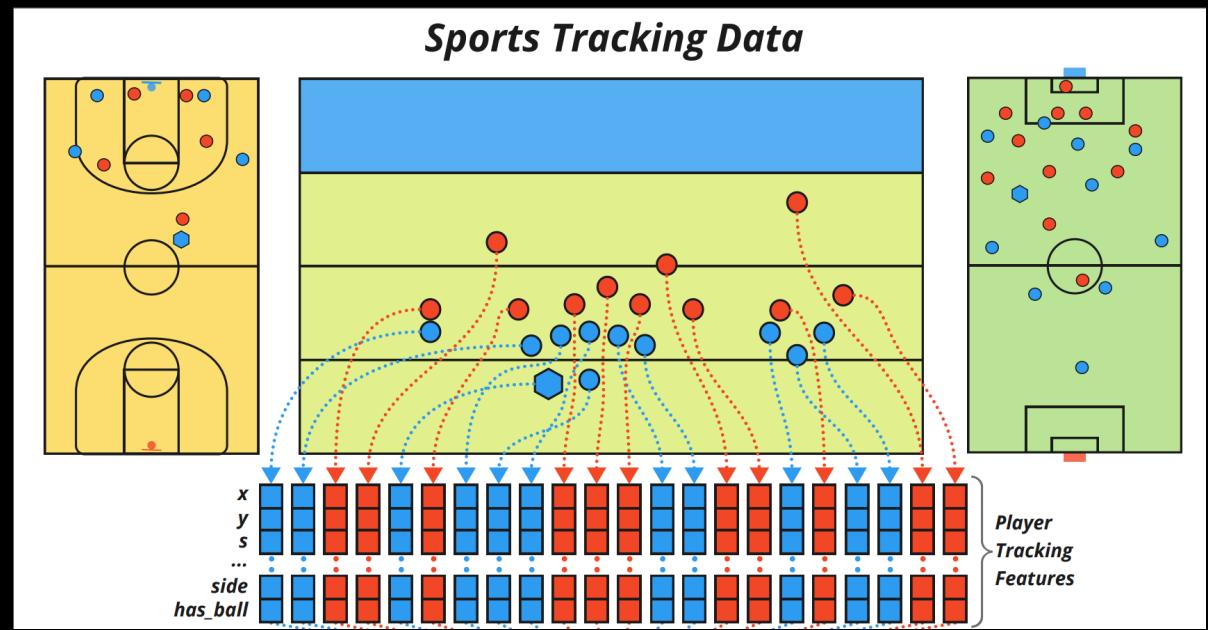


Credit: ChatGPT 4o ImageGen

Transformers and Attention!!!

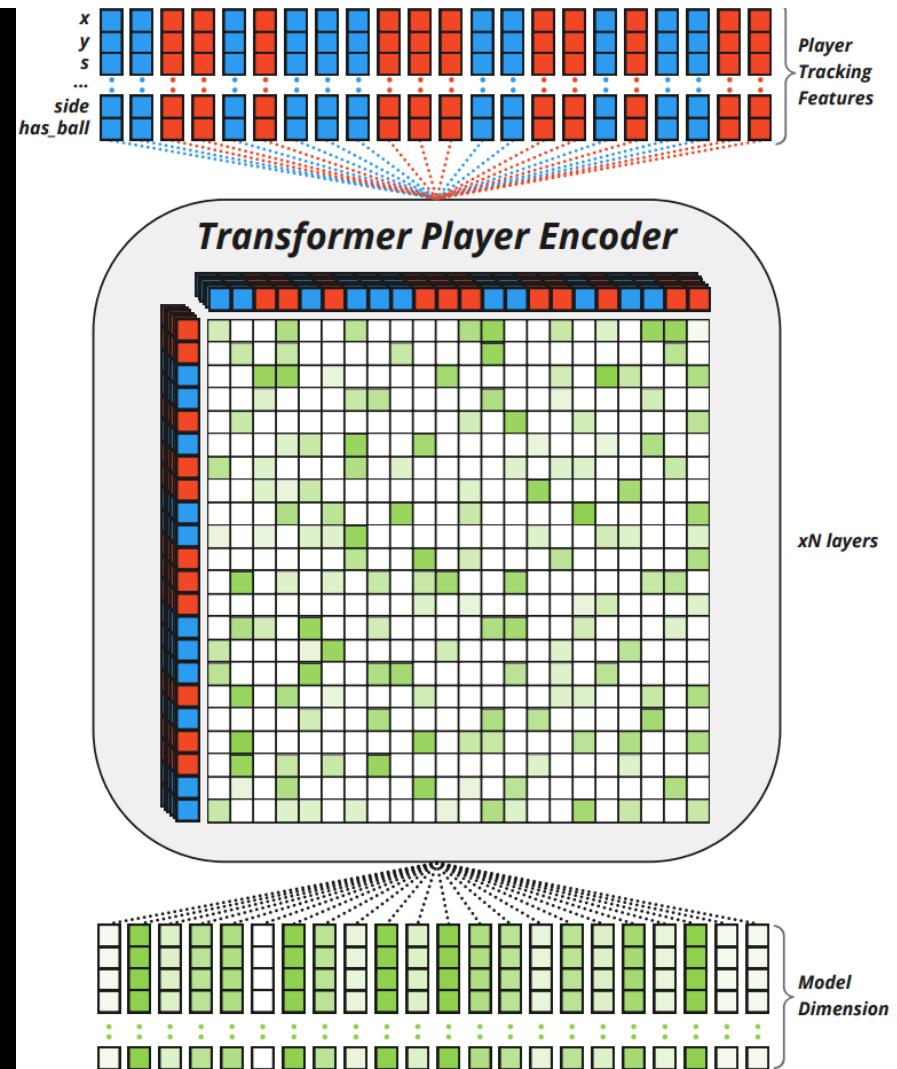
## Sports Tracking Transformer: Vector Creation

- Each player is an item in the “unordered bag”
- Spatial features go into respective item starting vector
- Any player ordering!
- No feature engineering needed
- Same for all sports or targets



## SportsTrackingTransformer: Stacking Transformer Layers

- Input and output of Transformer is the same data shape
- Chain these Transformer layers together
- Each layer gets maximal access to all other player vectors
- No collapsing like with pooling!
- Final output is a set of player vectors with salient values based on training objective



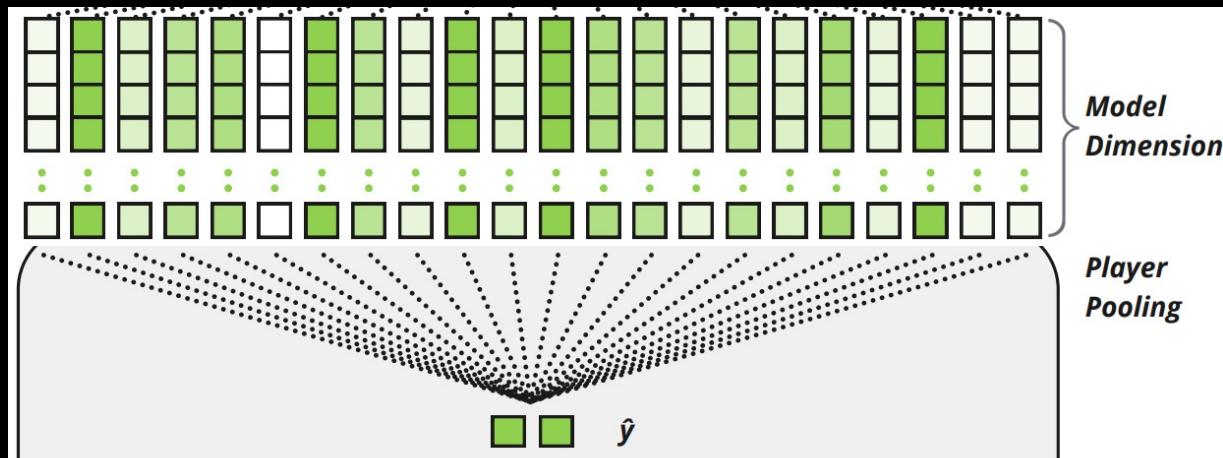
## Sports Tracking Transformer: Self-Attention

- Use each player vector to figure out attention weights between players
  - Columns are normalized to 1
  - Rows can give clues about “importance” of that player
- Attention weights are used to update the original vectors
- Re-ordering players simply re-orders the outputs
  - The data in the outputs for a given player will remain unchanged

	C. Jones	L. Dickerson	N. Bolton	S. Barkley	J. Hurts	D. Smith	J. Reid	T. McDuffie
C. Jones	0.1	<b>0.6</b>	0.1	0.2	0.0	0.0	0.1	0.0
L. Dickerson	<b>0.5</b>	0.1	0.0	0.1	0.0	0.0	0.1	0.0
N. Bolton	0.1	0.0	0.1	<b>0.3</b>	0.0	0.0	<b>0.3</b>	0.0
S. Barkley	<b>0.3</b>	<b>0.3</b>	<b>0.5</b>	0.3	<b>0.4</b>	0.0	<b>0.4</b>	0.0
J. Hurts	0.0	0.0	0.1	0.0	<b>0.6</b>	0.0	0.0	0.0
D. Smith	0.0	0.0	0.0	0.0	0.0	<b>0.2</b>	0.0	<b>0.8</b>
J. Reid	0.0	0.0	<b>0.2</b>	0.1	0.0	0.0	0.1	0.0
T. McDuffie	0.0	0.0	0.0	0.0	0.0	<b>0.8</b>	0.0	0.2

The diagram illustrates the self-attention process. An 8x8 attention matrix is shown, where rows and columns represent different players. The matrix entries are attention weights, with values ranging from 0.0 to 1.0. The columns are normalized to sum to 1.0. The rows provide information about the importance of each player. Below the matrix, arrows point from each row to its corresponding output bar. The output bars are color-coded: yellow for C. Jones, L. Dickerson, N. Bolton, S. Barkley, J. Hurts, D. Smith, J. Reid, and T. McDuffie. This visualizes how the data for a specific player remains unchanged when the order of the input sequence is altered.

## SportsTrackingTransformer: Getting Outputs



- Need to get prediction from final player vectors
- Last Step: Pool across players to finally collapse that dimension
- Tiny FFN to convert to data shape of target variable
  - Slight modifications for regression vs classification problems

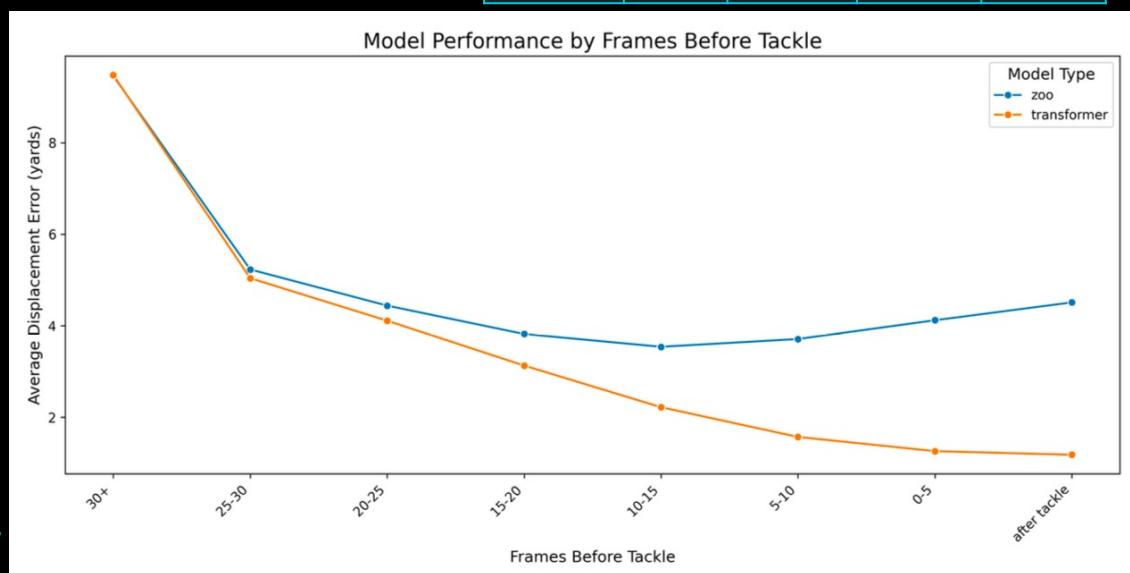
## Ok, but that's all theory...

- Bakeoff to predict tackle location
  - Close cousin to the objective of rushing yards in 2020
- Greatly expanded the dataset
  - 2024 BDB
  - Rushing and Run after Catch
  - All frames, not just handoff
- Performance
  - Very similar at ball snap and handoff
  - Transformer generalizes much better as play progresses

Split	%	Plays	Frames
Train	70	8.7k	740k
Val	15	1.9k	160k
Test	15	1.9k	160k

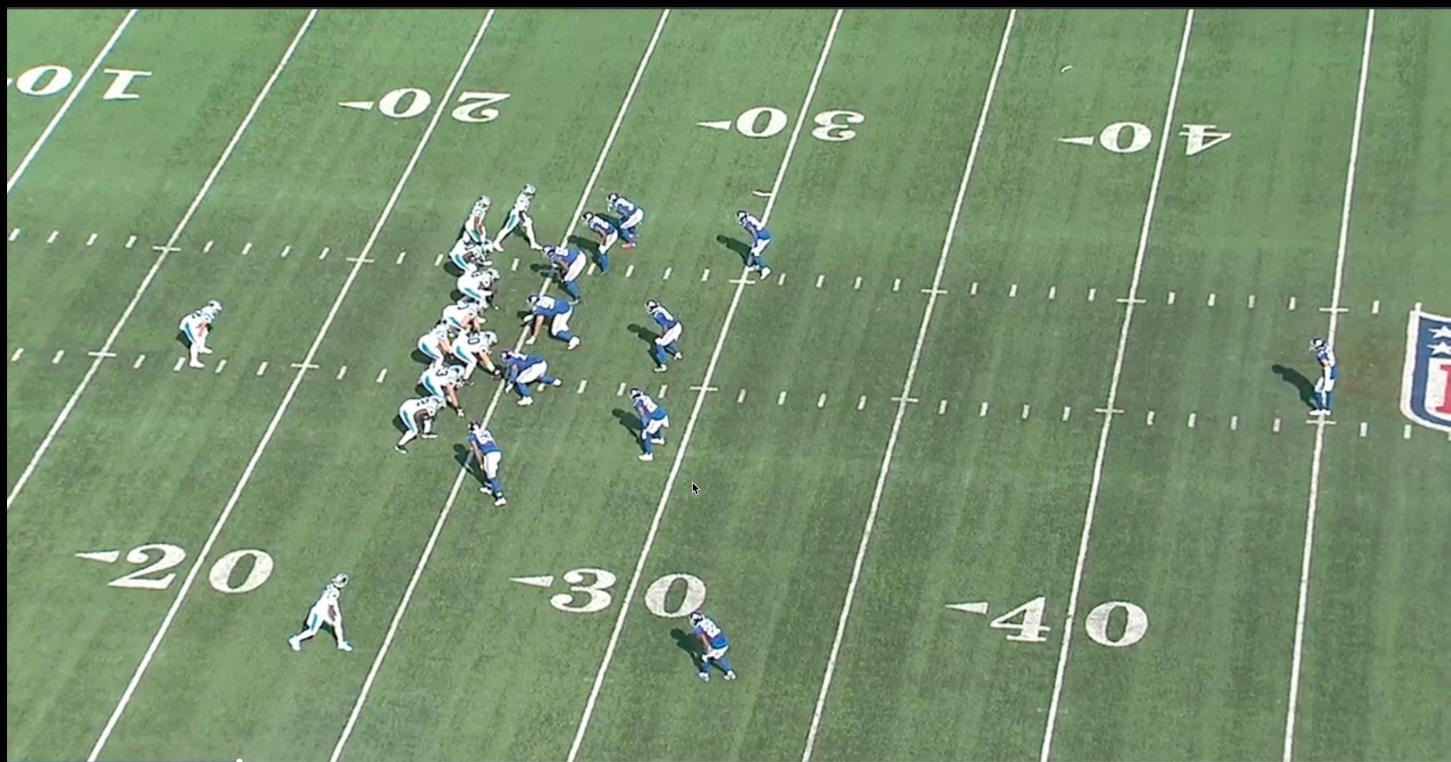
Test Set: Average Displacement Error in Yards

Event	n	Zoo	Transformer	Diff
Ball snap	958	8.64	8.90	-0.26
Handoff	887	6.63	6.51	0.12
Pass caught	842	4.46	4.17	0.29
First contact	1578	3.96	2.9	1.06
Tackle	1497	4.03	1.02	3.01



[Public GitHub: SumerSports/SportsTrackingTransformer](#)

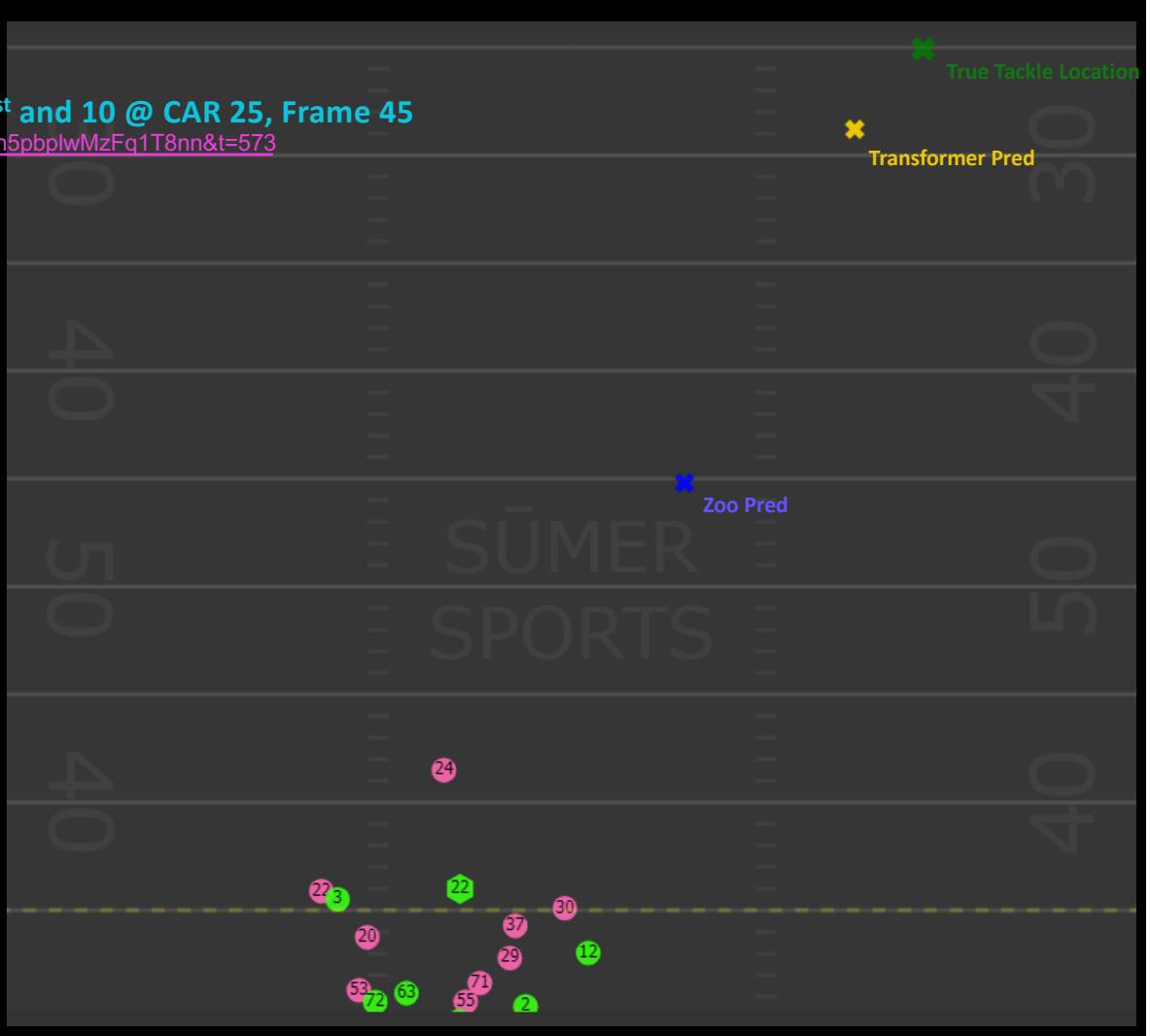
**Example:** Test Set: CAR vs NYG, 1<sup>st</sup> and 10 @ CAR 25, Frame 45  
[https://youtu.be/stsa\\_Llesak?si=n5pbplwMzFq1T8nn&t=573](https://youtu.be/stsa_Llesak?si=n5pbplwMzFq1T8nn&t=573)



## Example

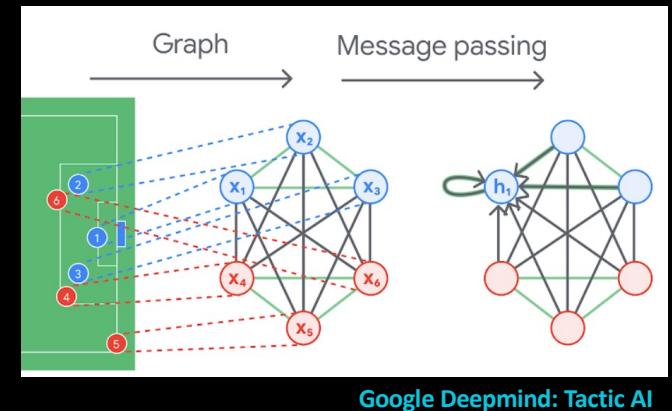
Test Set: CAR vs NYG, 1<sup>st</sup> and 10 @ CAR 25, Frame 45

[https://youtu.be/stsa\\_Llesak?si=n5pbplwMzFq1T8nn&t=573](https://youtu.be/stsa_Llesak?si=n5pbplwMzFq1T8nn&t=573)



## Other Invariant Architectures

- Set-based Learning (Horton 2020, Zaheer 2017)
  - Uses an invariant architecture, before Transformers became a powerful standard
- Graph Neural Nets (Wang, Veličković, et al 2024)
  - Tracking Data, like Natural Language, are a special case of graphs
    - Quite small (~25 nodes)
    - Dense (fully connected)
    - Few or No edge features
  - For sports, we think modern GNNs with attention and SportsTrackingTransformer are very similar
  - Alcorn and Nguyen [2021] found Transformers outperformed a GNN empirically for Basketball
- Invite empirical comparisons to these options!



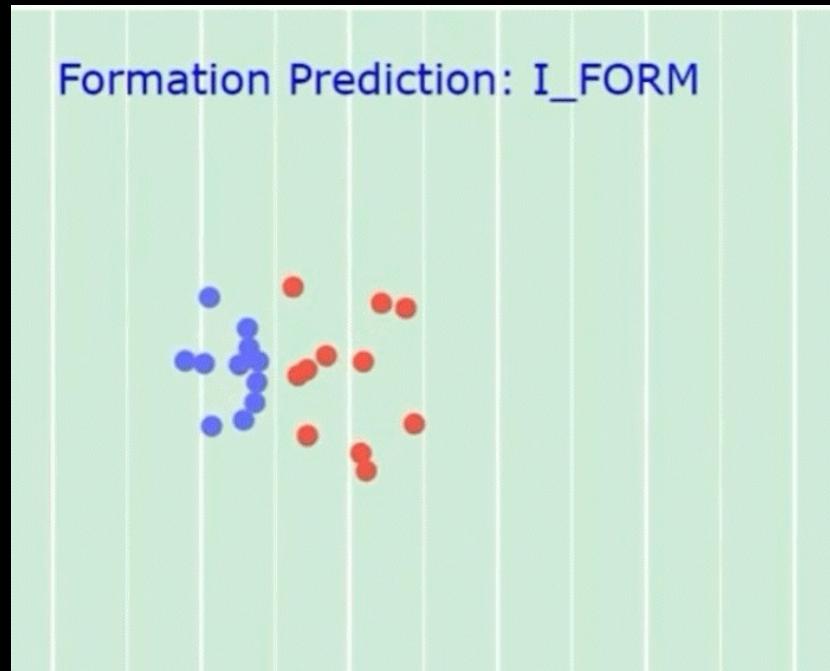
Google Deepmind: Tactic AI



Can the Transformer be viewed as a special case of a Graph Neural Network (GNN)?

## NFL BigDataBowl 2025 starter notebook

- Set up SportsTracking Transformer on Kaggle
- Trained on only 1 week of data to predict offensive formation



<https://www.kaggle.com/code/pvabish/modeling-with-transformers-by-sumersports>

# Code Walkthrough

[SumerSports/SportsTrackingTransformer](#)



# **2025 BDB: Pre-Snap Coverage Tells**



## How we started

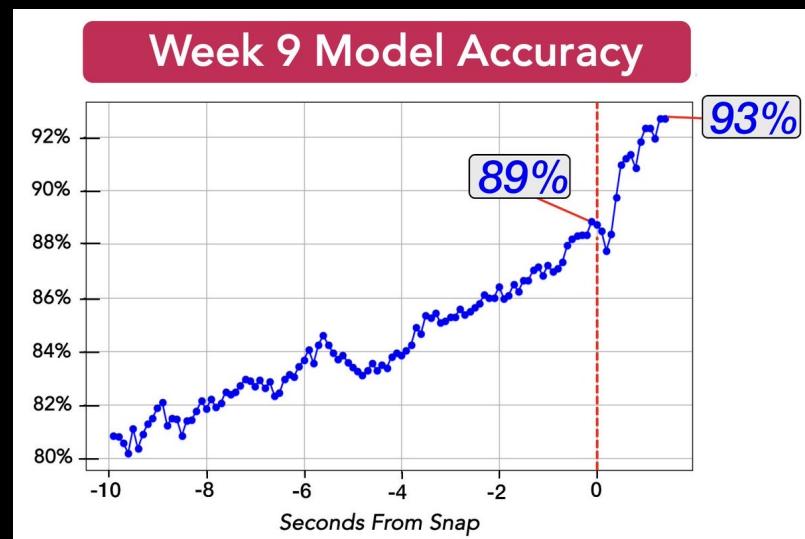
Getting our hands dirty

- Notebook to idea
  - Just tried implementing it
- Trying different ways to make sense of the multi-dimensional data
- Build confidence with tracking data and modeling code



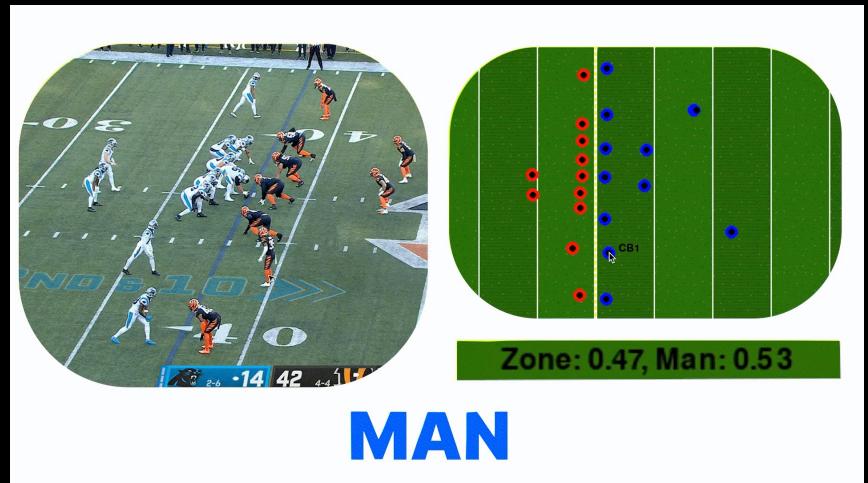
## Connecting To Football Problem

- Coaches/schemes/coverages all possible ideas
- Forcing our football knowledge onto attention matrix
- Picked Pre-Snap Man/Zone
  - Me and Smit worked from the architecture to the problem
  - Pooling layer has rich interactions -> Man/Zone
  - Known ties to Football



## Conveying to Football People

- Needs to be useful to NFL
  - Wanted an interactive tool
  - Having E2E models opens the door for creativity
- Audience is teams:
  - "Digital Whiteboard" – very team friendly
  - Simply hosting the data + model on DB + API



<https://x.com/VishakhS74/status/1896685773640069558>

# Adoption in the NFL



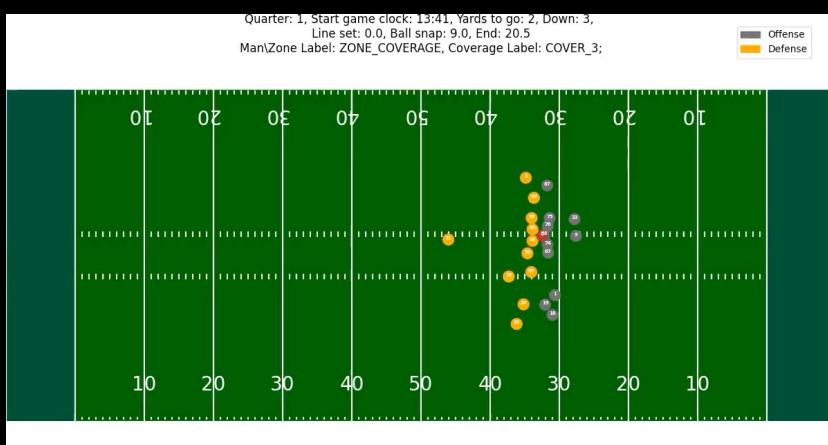
## NGS Coverage ID

- Me and Smit worked with NGS to create Coverage ID on prime vision
- Manning Cast on Mondays were tripped up by a play that the model got!



# Machines 1; Mannings: 0

Coverage ID



Manning Cast



Credit: @schwartzsteins on X

## SumerSight

- SumerSports uses Transformers for our in-house charting service
- Validate blindly against our in-house human experts
- You can access this data via SumerBrain (open Beta)!

<https://sumersports.com/the-zone/inside-the-data-how-smerbrain-sees-the-game/>

The screenshot shows the SumerBrain Beta interface with a dark theme. At the top, there's a logo and the text "SumerBrain Beta". Below that, a search bar contains the query: "Show me pressure stats (pressures, sacks, hurries, pressure rate) for the top 10 players in 2025 on plays where they align at DE or EDGE. Do not include off ball LB or interior DL alignments. Include their season position in the table, and how many pass rush snaps aligned at DE or EDGE". Underneath the search bar, the title "Top 10 Edge Rushers by Pressures in 2025 (DE/EDGE Alignment Only)" is displayed. A table follows, listing the top 10 players with their statistics. At the bottom, a note states: "Aidan Hutchinson leads the league with 42 pressures when aligned as a DE/EDGE, followed closely by Micah Parsons with 39 (these stats only include plays where they were aligned at DE or EDGE positions, excluding off-ball linebacker and interior defensive line alignments). Parsons has the highest pressure rate at 22.0% despite having fewer pass rush snaps than Hutchinson. Byron Young has been particularly efficient at converting pressures into sacks with 9.0 sacks from 30 pressures." A small dropdown arrow icon is located at the bottom right of the note.

Player	Position	Pass Rush Snaps	Pressures	Hurries	Sacks	Pressure Rate
Aidan Hutchinson	DE	251	42	23	6.0	16.7%
Micah Parsons	OB	177	39	22	5.5	22.0%
Tuli Tuipulotu	OB	198	37	20	5.0	18.7%
Jared Verse	DE	224	32	14	4.0	14.3%
Josh Hines-Allen	DE	209	31	20	0.5	14.8%
Will Anderson	DE	154	30	19	4.0	19.5%
Byron Young	DE	184	30	6	9.0	16.3%
Nik Bonitto	OB	142	29	10	8.0	20.4%
George Karlaftis	DE	191	29	11	3.5	15.2%
Jonathon Cooper	OB	149	27	15	6.0	18.1%

# Questions?



# Contact



**Udit Ranasaria**

[Udit.Ranasaria@sumersports.com](mailto:Udit.Ranasaria@sumersports.com)  
<https://x.com/uditranasaria>  
<https://www.linkedin.com/in/udittrana/>



**Vishakh Sandwar**

[Vishakh.sandwar@sumersports.com](mailto:Vishakh.sandwar@sumersports.com)  
<https://x.com/VishakhS74>  
<https://www.linkedin.com/in/vishakhs/>



**Pavel Vabishchevich**

[Pavel.vabishchevich@sumersports.com](mailto:Pavel.vabishchevich@sumersports.com)  
[https://x.com/Pavel\\_Vab](https://x.com/Pavel_Vab)  
<https://www.linkedin.com/in/pavel-vabishchevich-448b7673/>

