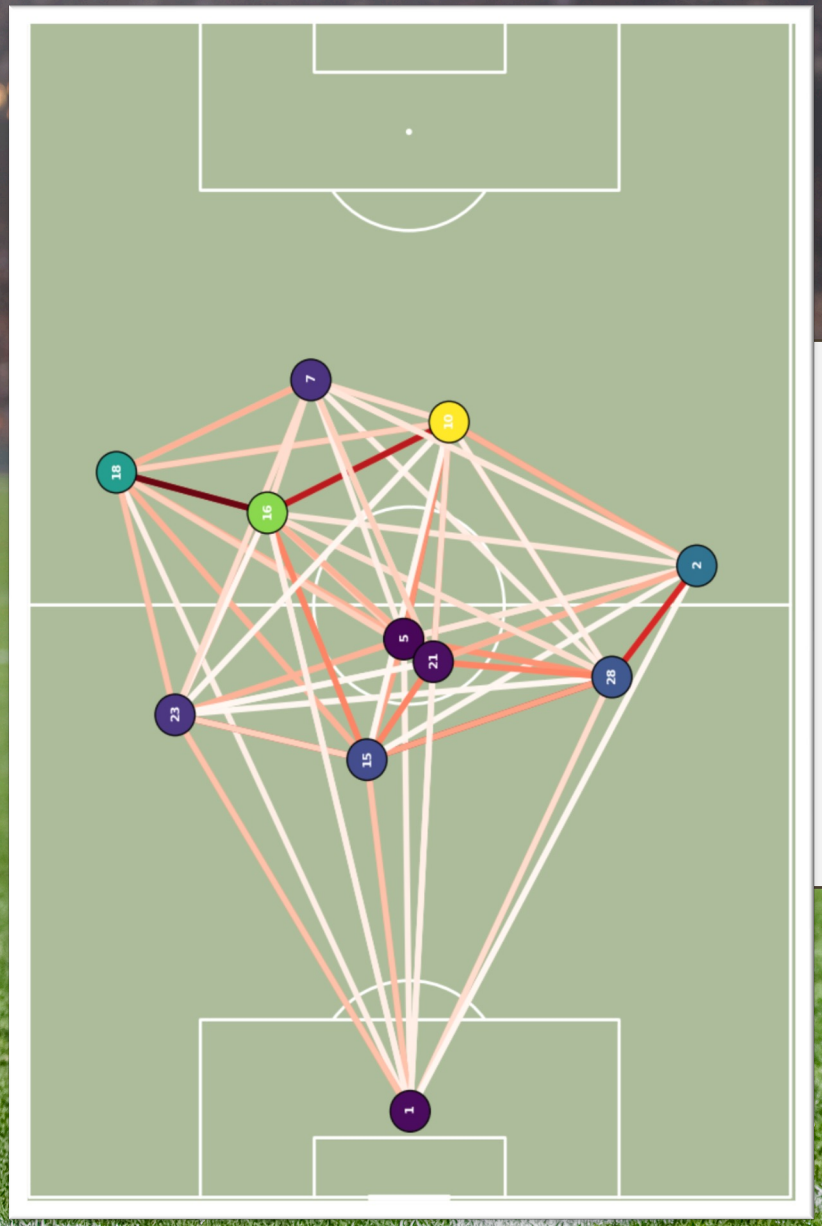


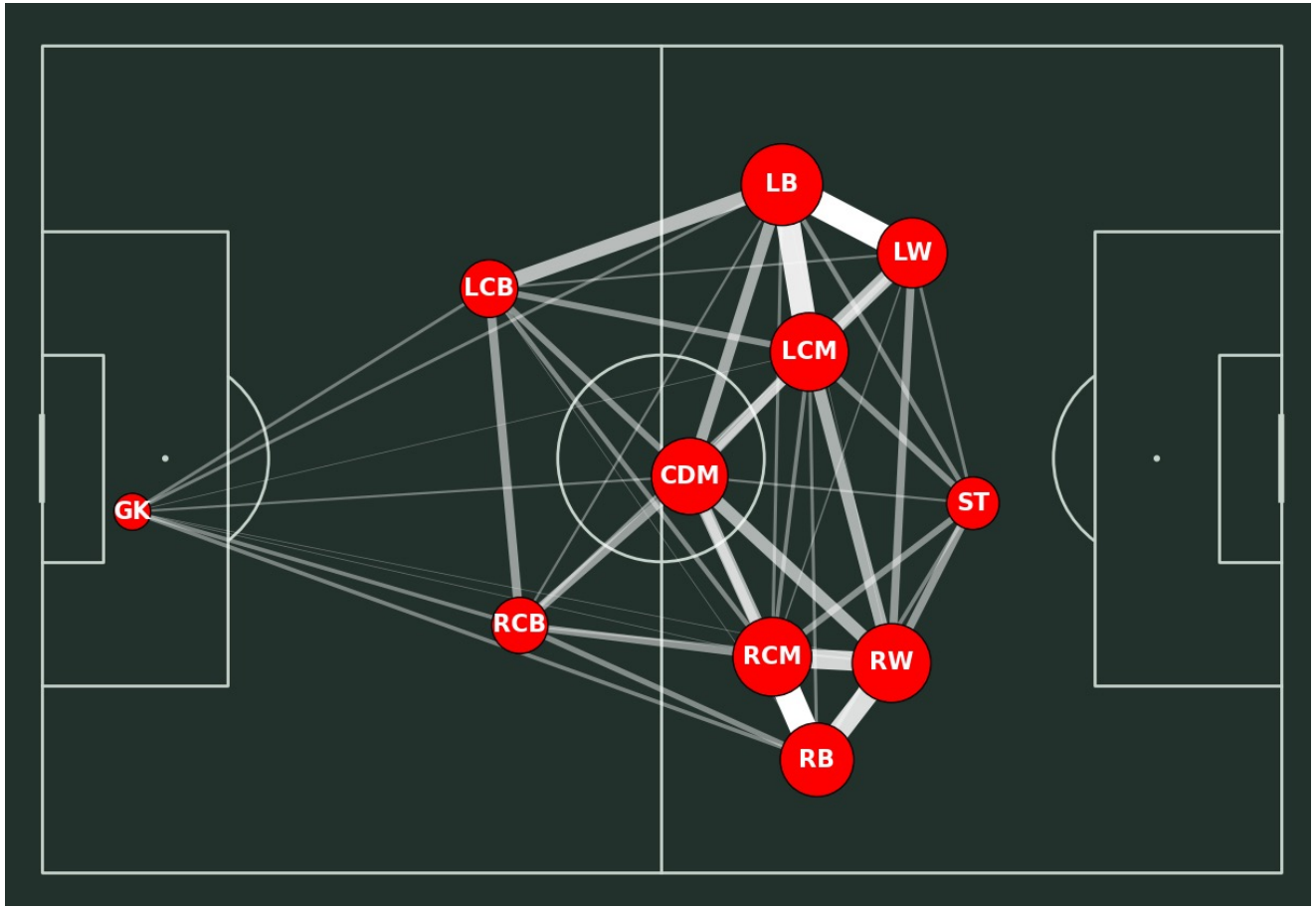
Understanding Soccer Through Data Science

Project by Atif Siddiqui

AGENDA

1. INTRODUCTION
    - WHAT ARE PASSING NETWORKS ?
    - GRAPHICAL NUERAL NETWORKS
    - ANALYSING PASSING NETWORKS USING GNN
2. PROJECT GOAL
3. DATA SPECIFICATION
4. HOW TO BUILD A PASSING NETWORK
5. A SIMPLE GNN ARCHITECTURE FOR CLASSFYING PASSING NETWORK
6. LSTM + GNN
7. EVALUTION
8. EURO 2024 FINAL ANALYSIS
9. FUTURE WORK
10. CONCLUSION

Passing Networks



What is a Passing Network?

It’s the application of network theory and social network analysis to passing data in soccer. Each player is a node, and the passes between them are connections.

Graphical Neural Network

What are Graph Neural Networks (GNNs)?

Graph Neural Networks (GNNs) are a class of neural networks designed to perform inference on data represented as graphs. Unlike traditional neural networks, which operate on grid-like structures (e.g., images), GNNs can process data that comes in the form of nodes (vertices) connected by edges (links).

Key Components:

Nodes: Represent entities in the graph.

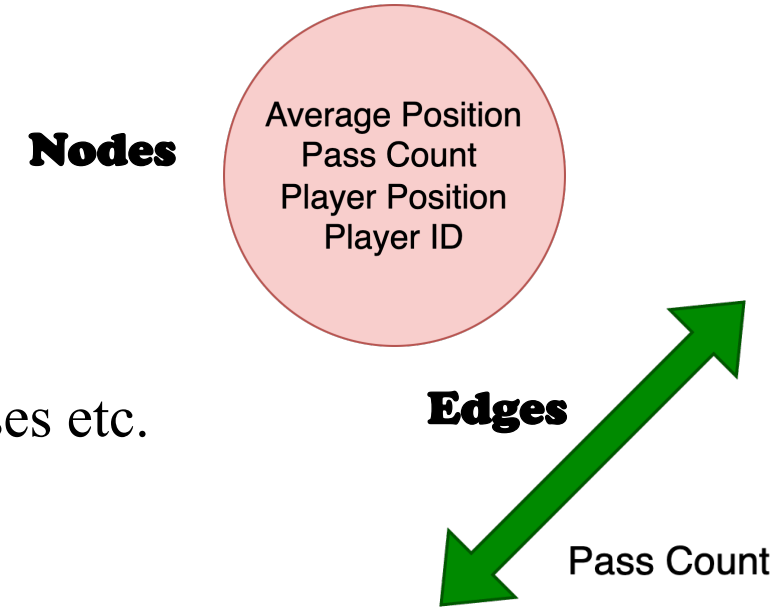
Edges (Links): Represent relationships or interactions between nodes.

Graph Structure: Defines the connectivity and relationship between nodes and edges.

Analyzing a Soccer Passing Network with GNNs

Data Representation:

1. **Nodes:** Players
2. **Edges:** Passes between players
3. **Node Features:** Average position on the pitch, total passes etc.



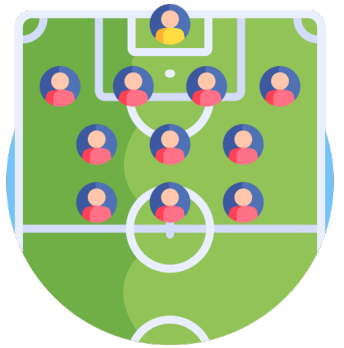
GNN Applications:

1. **Node Classification:** Classify players into different roles (e.g., defenders, midfielders, attackers) based on passing patterns.
2. **Link Prediction:** Predict potential successful passes in future games.
3. **Graph Classification:** Analyze team performance and classify match outcomes based on the passing network.

Goal of the Project

The goal of this project is to leverage Graph Neural Networks (GNNs) to classify soccer passing networks based on match outcomes, and to develop a new metric called PassNetScore. This metric will predict the probability of winning a match, ranging from 0 to 1, based on the structure and characteristics of the passing network.

Data



Statsbomb Event Data

- Provides detailed information for every event that occurs during a match.
- Includes a wide range of events such as passes, shots, fouls, substitutions, and more.

Passing Data:

- Each pass is documented with the player who made the pass and the recipient.
- Includes the outcome of the pass (e.g., complete, incomplete, intercepted).
- Captures the precise location and timing of each pass, enabling detailed analysis of passing patterns.

Data Specification

Match Outcomes:

- Each event is linked to the overall outcome of the match (win, loss, draw).
- This linkage allows for the analysis of how specific events and strategies correlate with match results.

Player Data

- Using event data, the average position of each player can be calculated.
- Player-specific metrics can be derived, such as pass accuracy, pass frequency, and involvement in key plays.

Data Specification

Graphs: 2096

Grouped Graphs: 24576

Data Snapshot

index	id	match_id	minute	type ▲	location	player	player_id	pass_outcome	pass_recipient	pass_recipient_id	team	team_id	shot_outcome	substitution_replacement	substitution_replacement_id	tactics	pass_en
9	e6b6543e-f9cc-4a64-b907-059a48a3a835	3773386	0	Pass	67.8,41.9	Deyverson Brum Silva Acosta	25134.0	Incomplete	José Ignacio Peleteiro Ramallo	9462.0	Deportivo Alavés	206	NaN	NaN	NaN	NaN	94.1,19.6
10	e3d8ed99-7ff2-4962-99bb-4ac922f2699c	3773386	0	Pass	42.4,12.9	Rubén Duarte Sánchez	6612.0	Complete	Deyverson Brum Silva Acosta	25134.0	Deportivo Alavés	206	NaN	NaN	NaN	NaN	66.3,41.7
29	d4c6d660-4d40-413a-80b2-0dd74b06461d	3773386	1	Pass	61.7,51.4	Deyverson Brum Silva Acosta	25134.0	Complete	Edgar Antonio Méndez Ortega	26387.0	Deportivo Alavés	206	NaN	NaN	NaN	NaN	94.1,67.5
30	11595aea-f7f3-4954-b46a-649783043289	3773386	1	Pass	51.2,63	Edgar Antonio Méndez Ortega	26387.0	Complete	Deyverson Brum Silva Acosta	25134.0	Deportivo Alavés	206	NaN	NaN	NaN	NaN	61.7,50.5
46	cba08114-7620-478a-9506-7303899b6ead	3773386	2	Pass	48.7,22.9	Tomás Pina Isla	6622.0	Complete	Florian Grégoire Claude Lejeune	3093.0	Deportivo Alavés	206	NaN	NaN	NaN	NaN	35.9,20.2
47	2c5823ca-b796-4329-9241-86f6a7c44fc5	3773386	2	Pass	50.1,18.4	Deyverson Brum Silva Acosta	25134.0	Complete	Tomás Pina Isla	6622.0	Deportivo Alavés	206	NaN	NaN	NaN	NaN	48.3,22.4
49	6fc63a1d-d1ee-4f0c-9b07-0a294b46ac08	3773386	2	Pass	49.2,2.3	Deyverson Brum Silva Acosta	25134.0	Incomplete	Luis Jesús Rioja González	24049.0	Deportivo Alavés	206	NaN	NaN	NaN	NaN	56.7,12.4
50	0980b42d-37b9-4a95-8a67-c9ce593fa9ce	3773386	2	Pass	35.9,20.2	Florian Grégoire Claude Lejeune	3093.0	Complete	Edgar Antonio Méndez Ortega	26387.0	Deportivo Alavés	206	NaN	NaN	NaN	NaN	61.7,74.3
55	8ad69aa7-688e-47e9-93ae-56e0574e7841	3773386	2	Pass	35.4,0.1	Rubén Duarte Sánchez	6612.0	Complete	Deyverson Brum Silva Acosta	25134.0	Deportivo Alavés	206	NaN	NaN	NaN	NaN	48.4,2.5

Building a Passing Network Graph

- Getting data from Statsbomb event data
- Filter passing data
- Build weighted adjacency matrix
- Construct player nodes with features
- Construct a graph using Spectral (A Python library for graph deep learning, based on the Keras API and TensorFlow 2)

```
Graph(n_nodes=11, n_node_features=3, n_edge_features=None, n_labels=1)
```

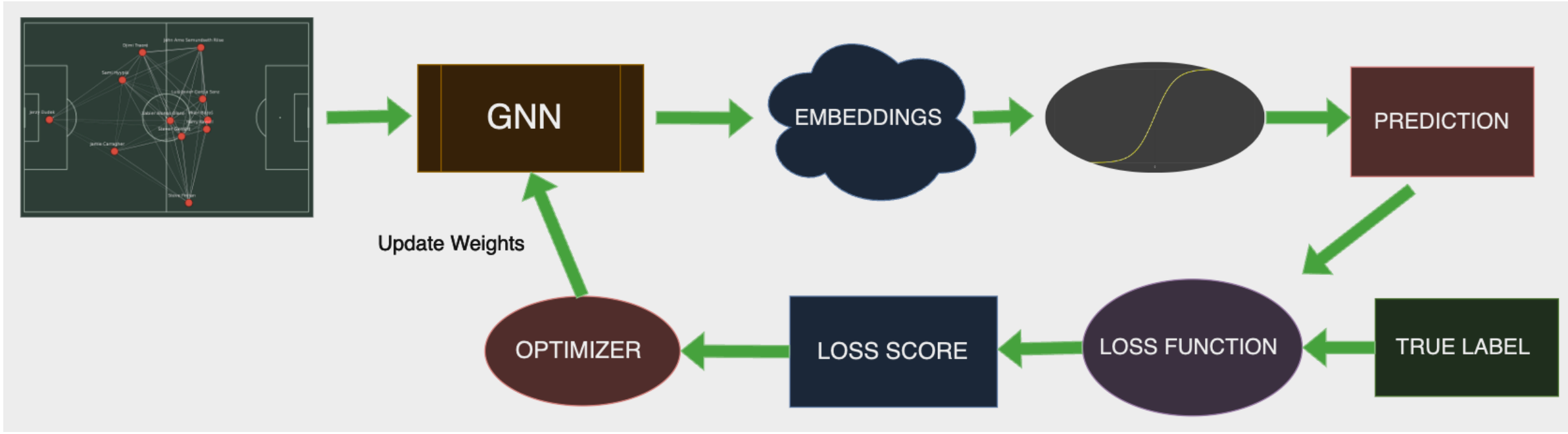
```python
from spektral.data import Dataset, Graph
import numpy as np
import scipy.sparse as sp
from spektral.utils import reorder
class CustomDataset(Dataset):
    """
    A dataset of custom graphs.
    """

    def __init__(self, graphs, **kwargs):
        self.graphs = graphs
        super().__init__(**kwargs)

    def read(self):
        def make_graph(graph_data):
            adjacency_list, node_features, edge_features, label = graph_data

            # Convert adjacency list to sparse adjacency matrix
            n_nodes = len(node_features)
            adj_matrix = np.zeros((n_nodes, n_nodes))
            # edge_matrix = np.zeros((n_nodes, n_nodes))
            for edge in edge_features:
                src, dst, value = edge
                adj_matrix[src, dst] = value

            # edge_matrix = np.zeros(edge_features)
            # print(edge_matrix)
            return Graph(
                x=node_features,
                a=sp.csr_matrix(adj_matrix),
                # e=edge_features,
                y=label
            )

        return [make_graph(graph_data) for graph_data in self.graphs]
```

A Simple Graph Neural Network Architecture

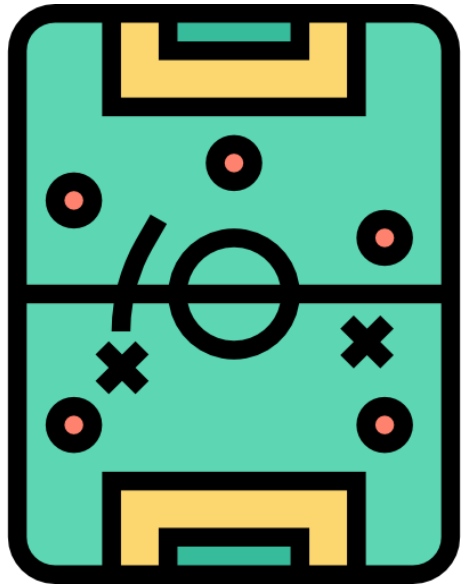
Average Position
Pass Count
Player Position
Player ID

Node

Pass Count or Edge Occurrence



A Simple GNN



This layer expects a sparse adjacency Convolutional Layer:
A crystal graph convolutional layer matrix.

This layer computes:

$$\mathbf{x}_i' = \mathbf{x}_i + \sum_{j \in \mathcal{N}(i)} \sigma\left(\mathbf{z}_{ij}\mathbf{W}^{(f)} + \mathbf{b}^{(f)}\right) \odot \mathbf{g}\left(\mathbf{z}_{ij}\mathbf{W}^{(s)} + \mathbf{b}^{(s)}\right)$$

where $\mathbf{z}_{ij} = \mathbf{x}_i \| \mathbf{x}_j \| \mathbf{e}_{ji}$, σ is a sigmoid activation, and g is the activation function (defined by the `activation` argument).

Input
Node features of shape (n_nodes, n_node_features);
Binary adjacency matrix of shape (n_nodes, n_nodes).
Edge features of shape (num_edges, n_edge_features).
Output
Node features with the same shape of the input.

A look inside the model

Learning rate: learning_rate = 1e-3

Hidden units for the neural network: channels = 128

Number of CrystalConv layers: layers = 3

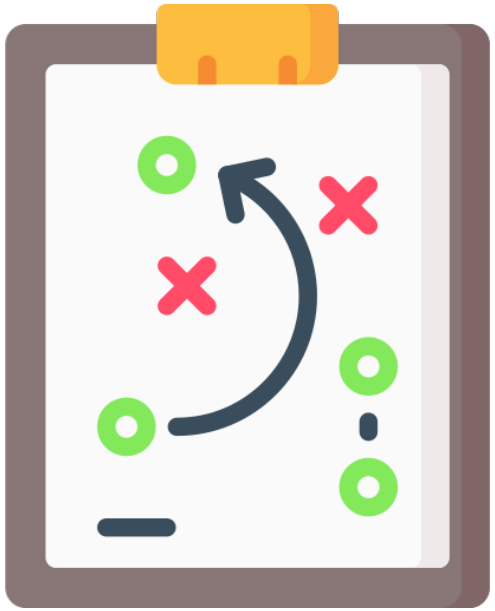
Activation Functions:

ReLU activation for intermediate layers and sigmoid activation for the final output.

Global Average Pooling: GlobalAvgPool()

Optimizer: Adam(learning_rate)

Loss Function: BinaryCrossentropy()

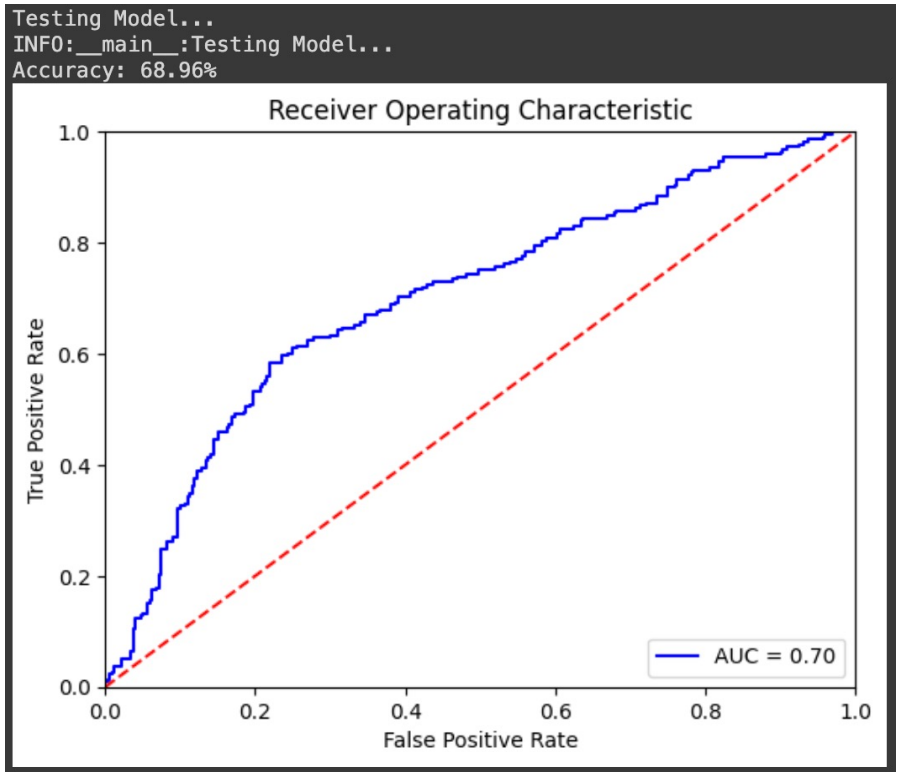


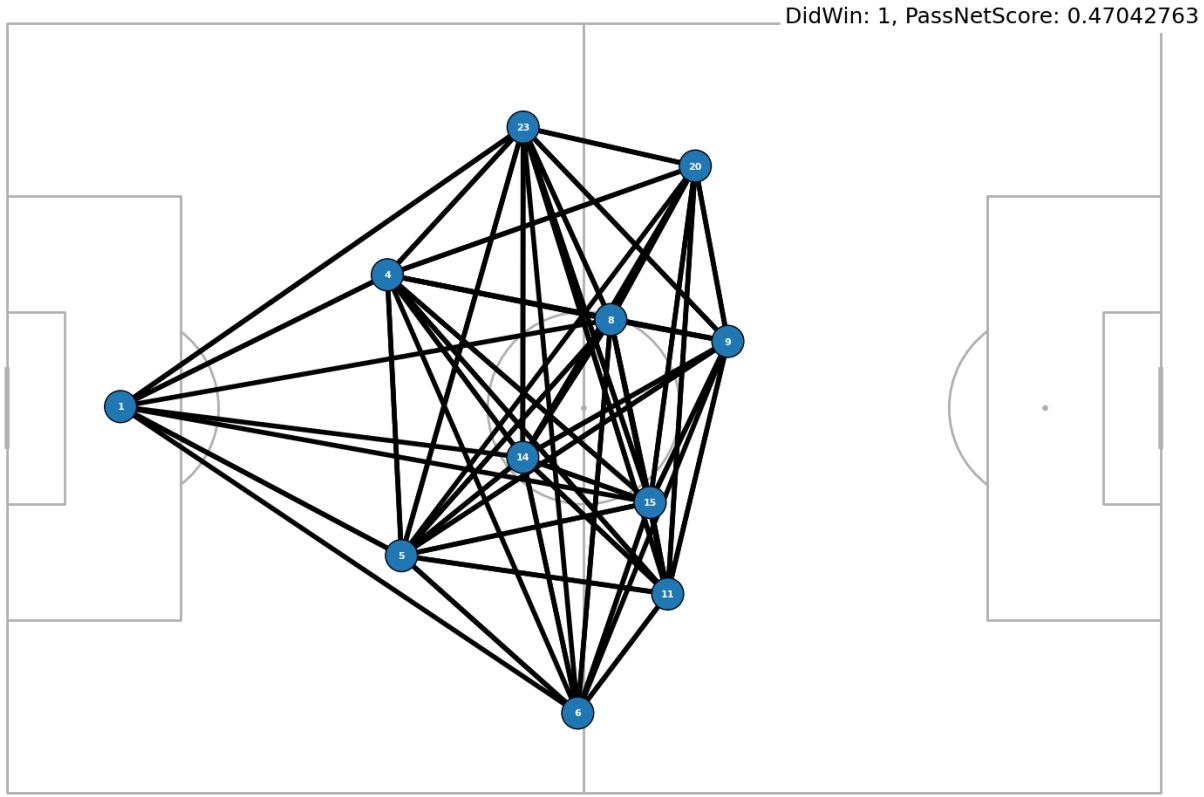
Model 1: Binary Adj Matrix

```
Graph(n_nodes=11, n_node_features=3, n_edge_features=None, n_labels=1)
```

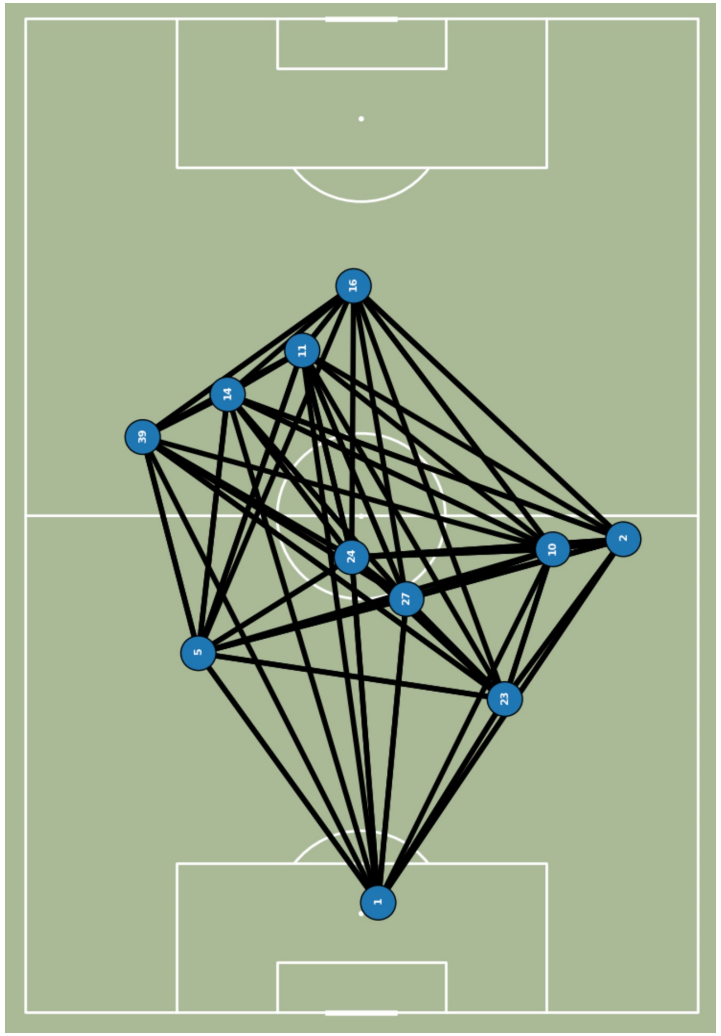
Average Position
Pass Count
Player Position
Player ID

Edge Occurrence
(0,1)

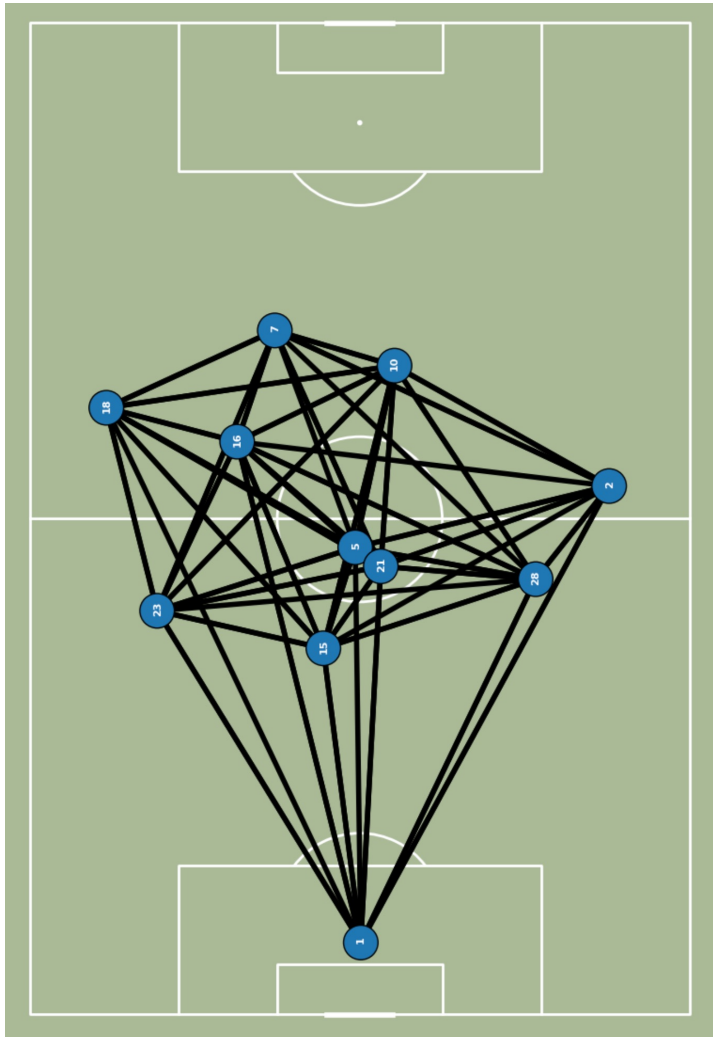




Model 1: Does it make sense ?



Match	Osasuna (0)	Barcelona (2)
Total Passes	358	674
Pass Completion %	83.9%	90.8%
Match Won	NO	YES
Goals Scored	0	2
PassNet Score	0.15384474	0.6673707



Osasuna Passing Network

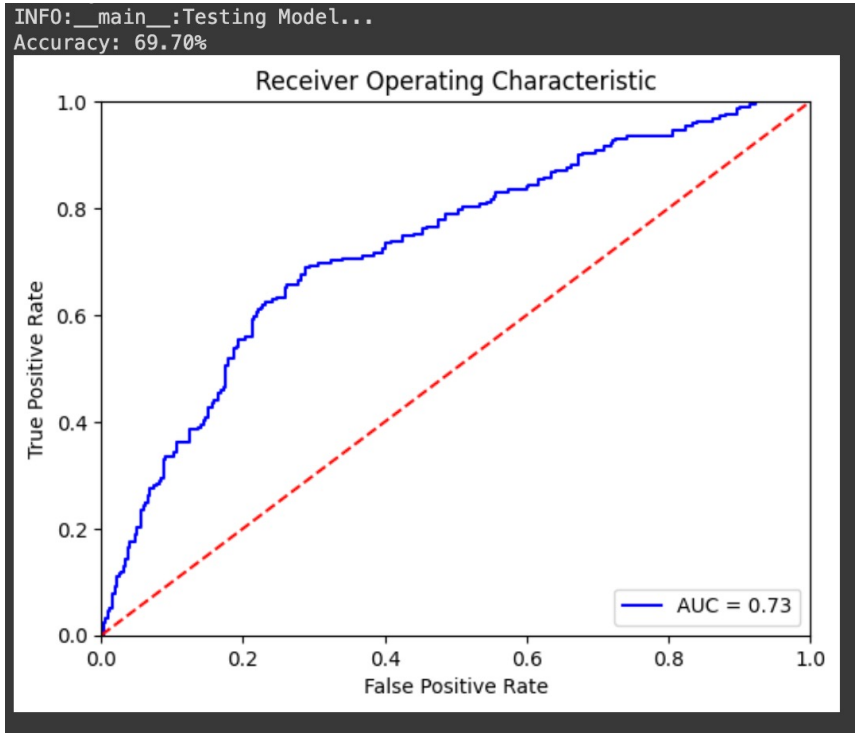
Barcelona Passing Network

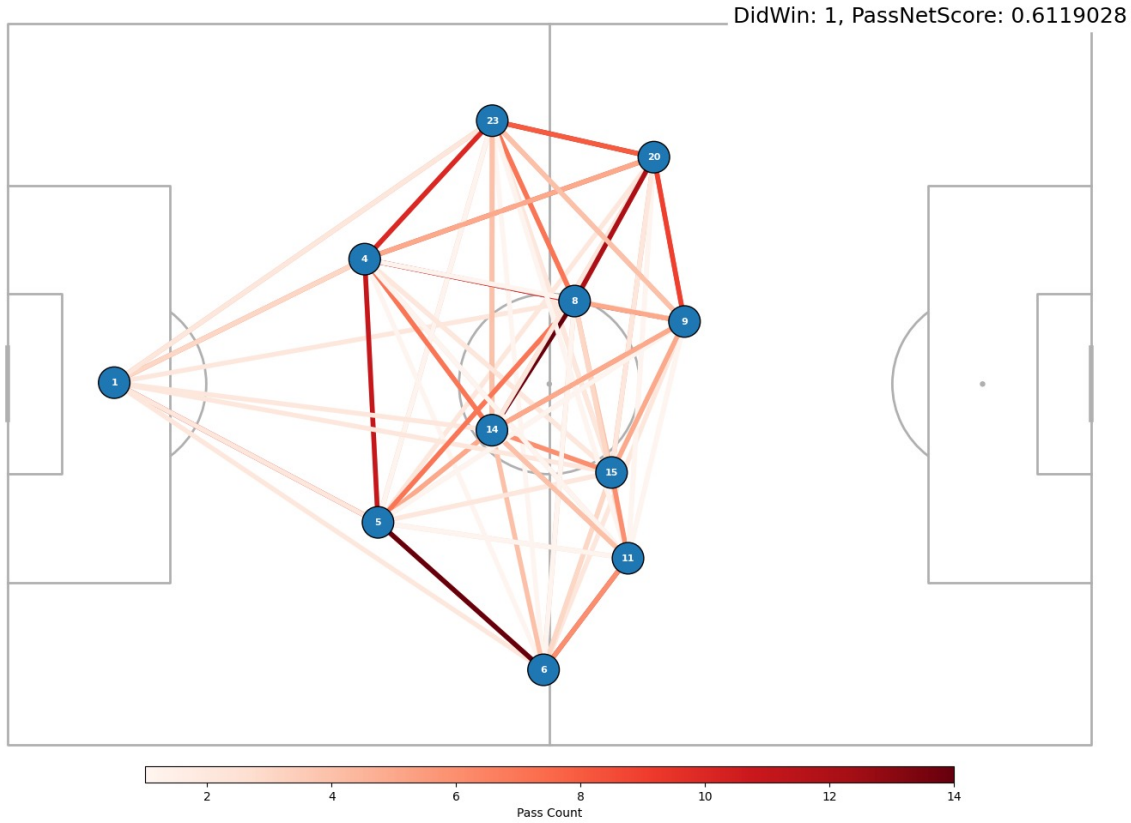
Model 1: Let's add more information

```
Graph(n_nodes=11, n_node_features=4, n_edge_features=1, n_labels=1)
```

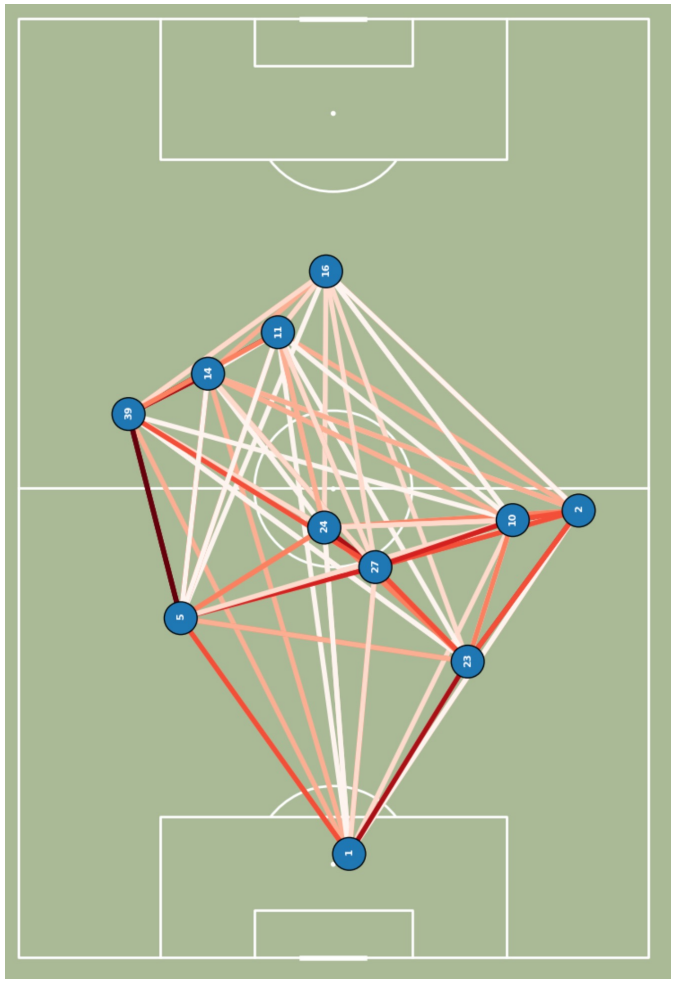
Average Position
Pass Count
Player Position
Player ID

Pass Count

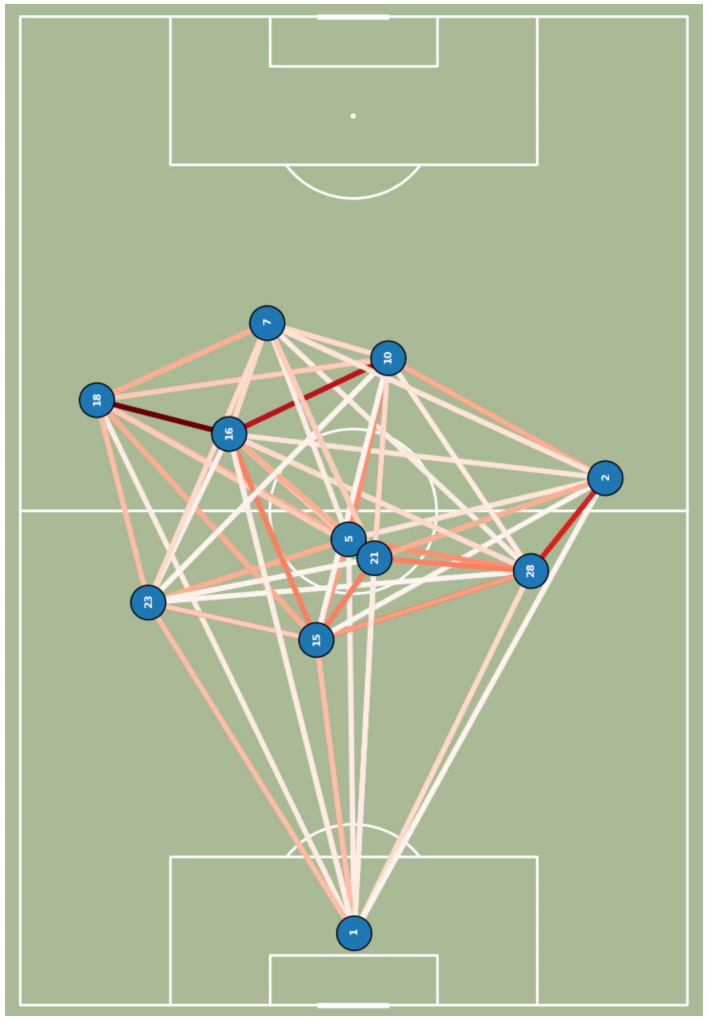




Model 1: Does it make sense ?



Match	Osasuna (0)	Barcelona (2)
Total Passes	358	674
Pass Completion %	83.9%	90.8%
Match Won	NO	YES
Goals Scored	0	2
PassNet Score	0.31328285	0.5977419



Osasuna Passing Network

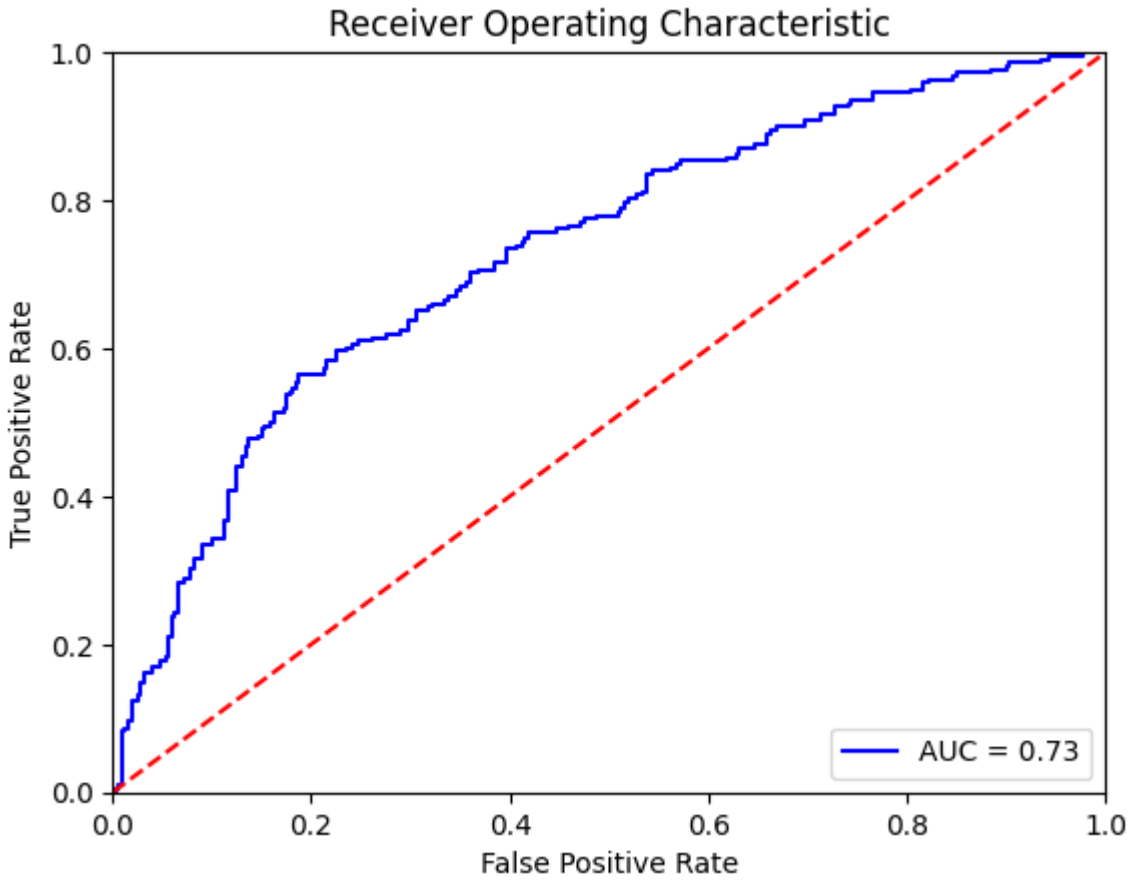
Barcelona Passing Network

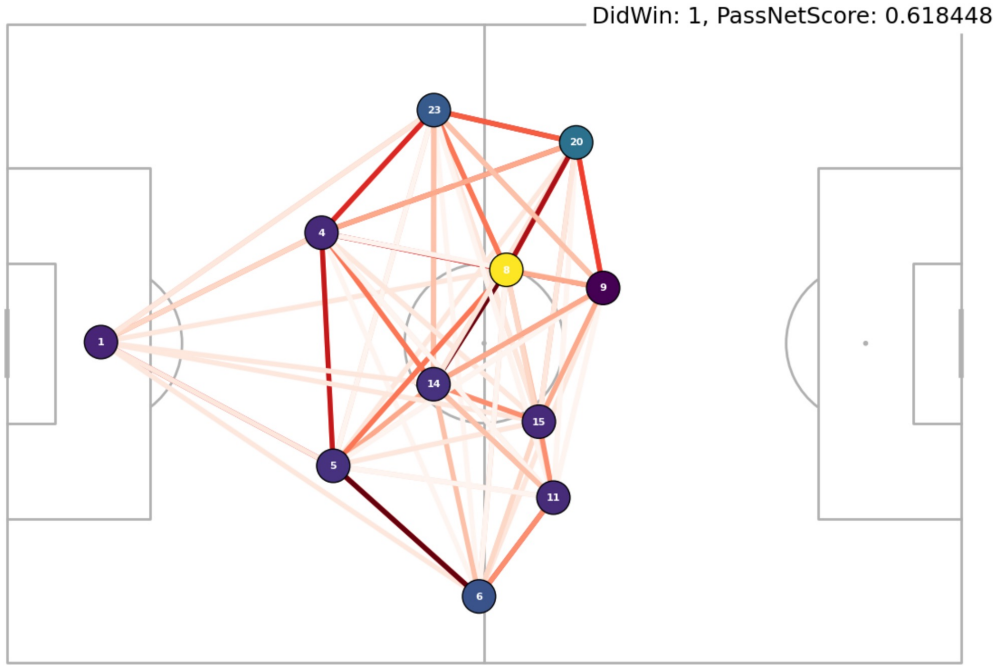
Adding Expected Threat

Average Position
Pass Count
Player ID
Position
XT

Pass Count

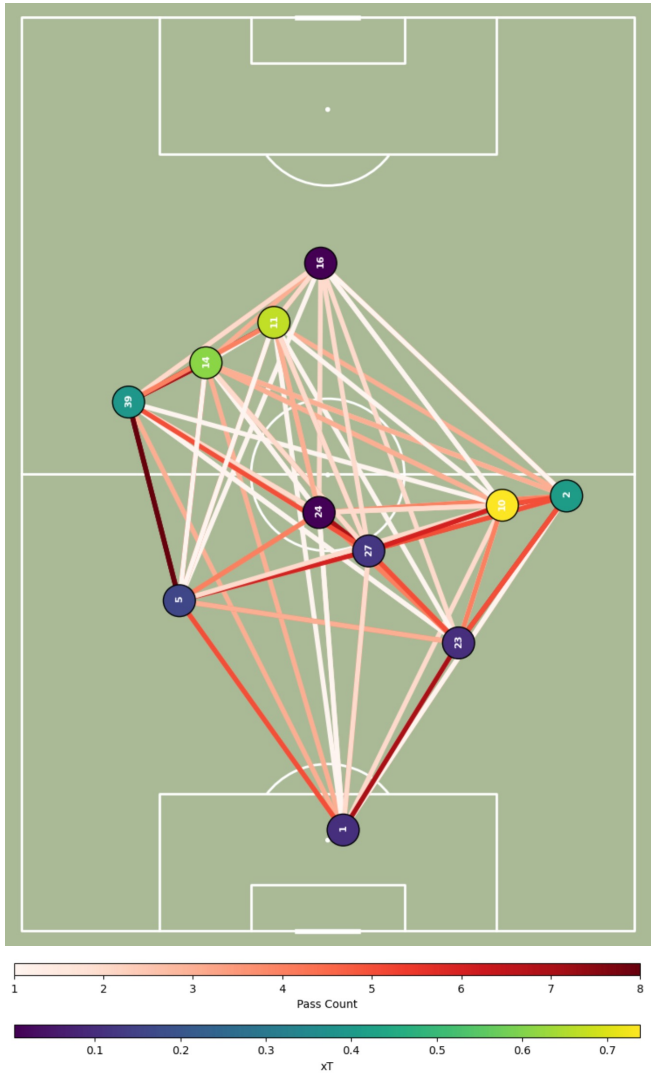
```
Graph(n_nodes=11, n_node_features=5, n_edge_features=1, n_labels=1)
```





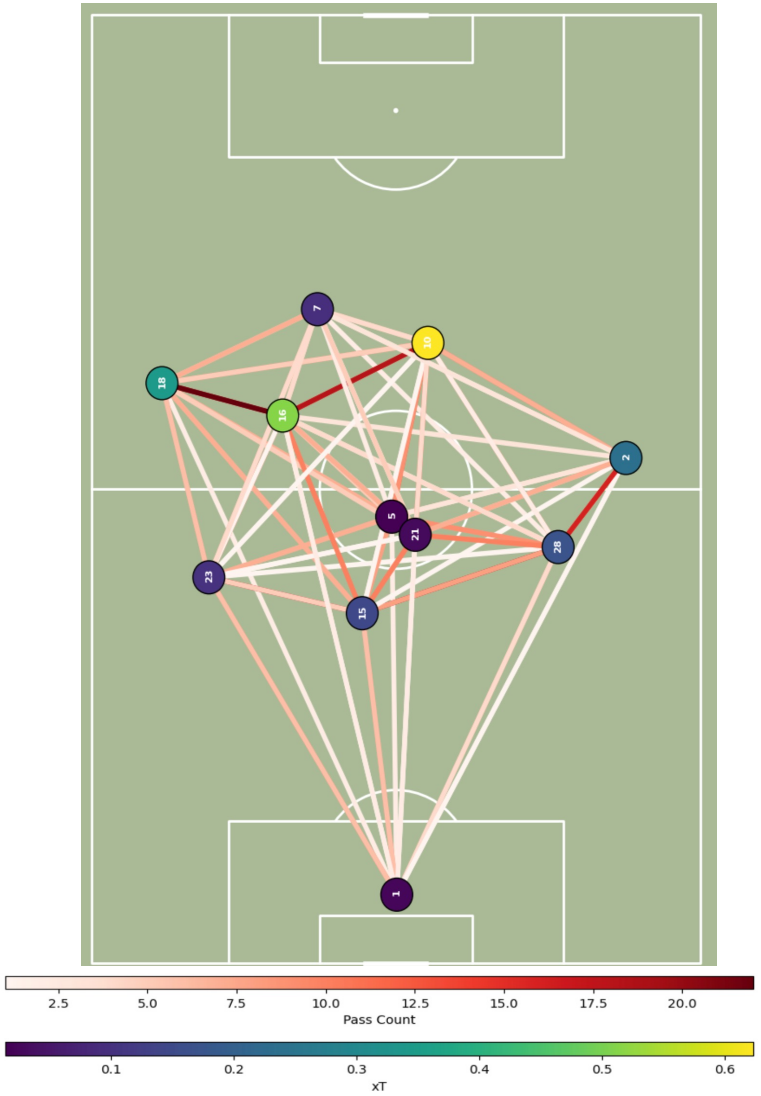
Model 1: Does it make sense ?

Osasuna Passing Network



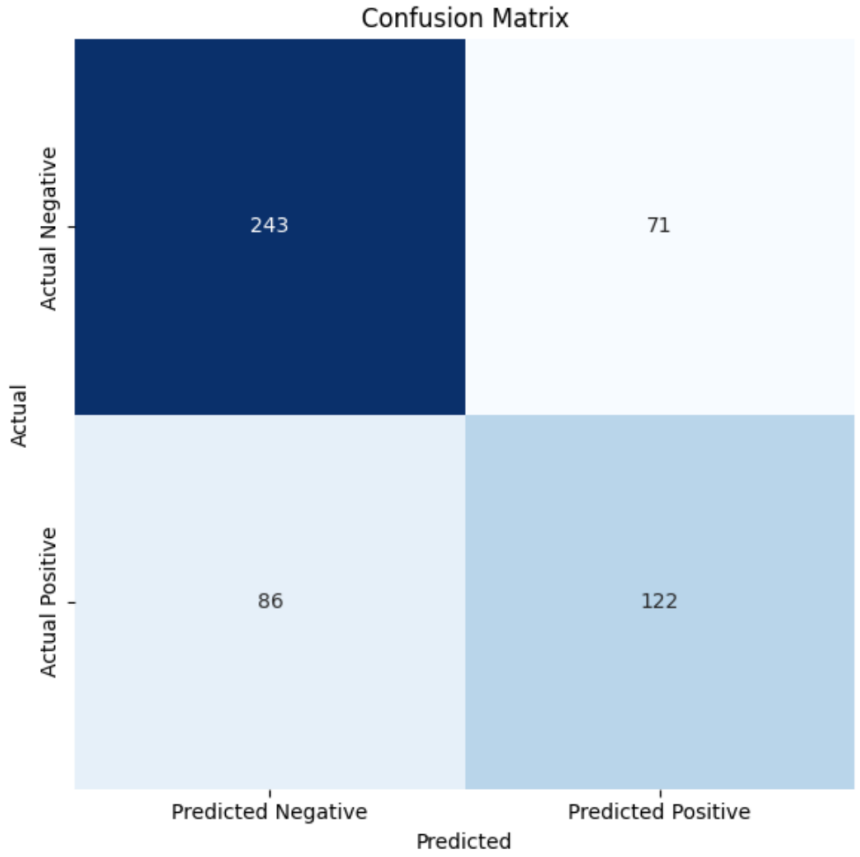
Match	Osasuna (0)	Barcelona (2)
Total Passes	358	674
Pass Completion %	83.9%	90.8%
Match Won	NO	YES
Goals Scored	0	2
PassNet Score	0.26198548	0.59394646

Barcelona Passing Network

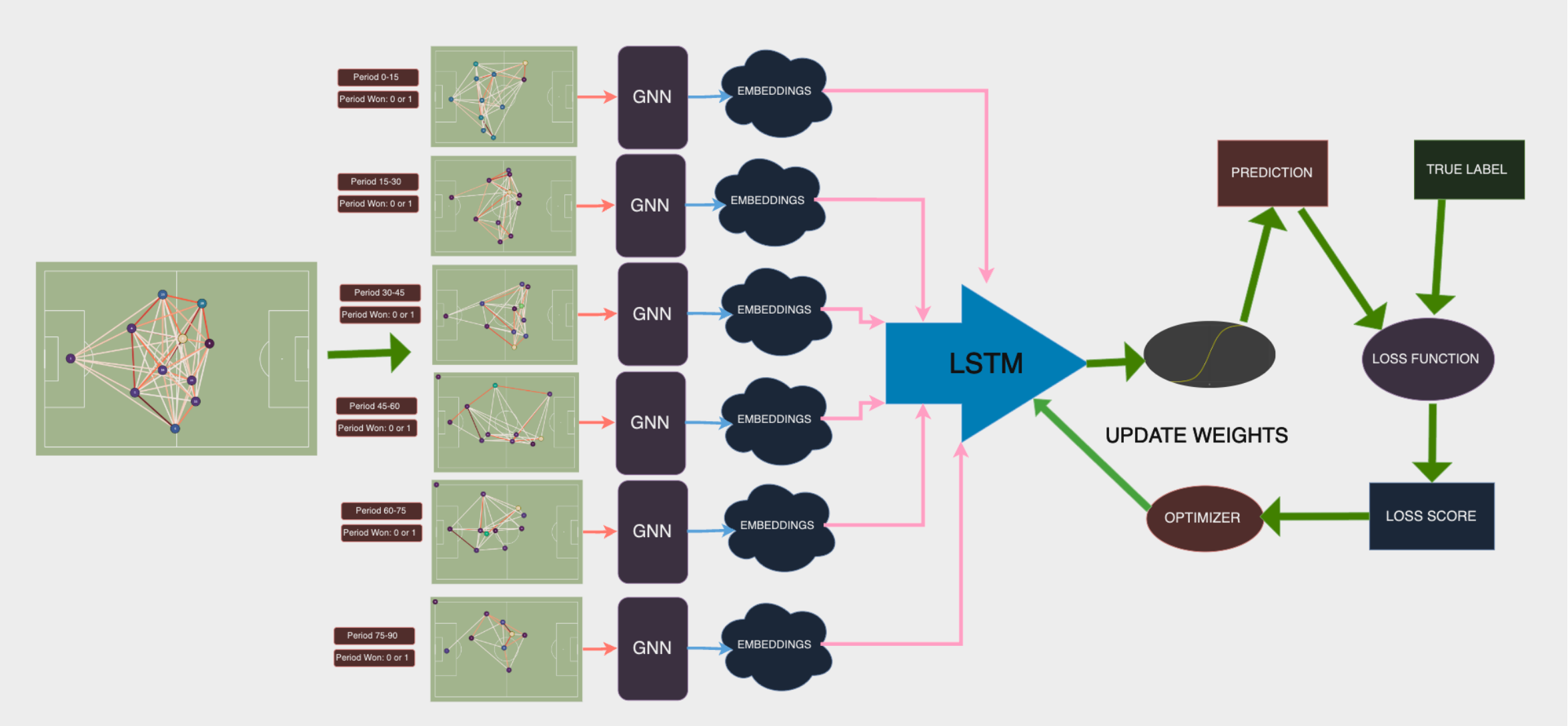


Model 1 Evaluation

Models	Binary	Pass Count	xT
Accuracy	0.68	0.69	0.72



Model 2: GNN + LSTM



A look Inside the GNN Model

Graph Convolutional Layers:

GTVConv: Custom graph convolution layer that computes node embeddings by aggregating features from neighboring nodes.

Parameters such as channels, delta_coeff, epsilon, and activation are used to configure the convolution operation.

Regularization and constraints are applied to control overfitting and improve generalization.

Readout Layer

Dense Layer: Transforms node embeddings to the specified number of readout units.

Global Max Pooling: Aggregates node embeddings into a single graph-level representation by taking the maximum value across all nodes for each feature.

Hyperparameters

Hyperparameters Summary

channels: Number of features for each graph convolution layer (default: 32).

delta_coeff: Coefficient for attention mechanism in GTVConv (default: 1.0).

epsilon: Small constant to avoid division by zero (0.001).

n_layers: Number of graph convolutional layers (3).

readout_units: Number of units in the readout dense layer (16).

activation: Activation function for layers (default: 'relu').

use_bias: Whether to use bias in the layers (True).

kernel_initializer: Initializer for kernel weights ('he_normal').

bias_initializer: Initializer for bias terms (default: 'zeros').

Hyperparameters

Hyperparameters Summary

kernel_regularizer: Regularizer for kernel weights (L1 regularization).

bias_regularizer: Regularizer for bias terms (None).

activity_regularizer: Regularizer for layer output (default: None).

kernel constraint: Constraint function for kernel weights (None).

bias_constraint: Constraint function for bias terms (default: None).

Additional Notes

Regularization: L1 regularization is used to encourage sparsity in the weights.
Constraints: Constraints such as MaxNorm are applied to kernel and bias to control the magnitude of the weights.
Pooling: Global max pooling is used to aggregate node features into a graph-level representation.

Node Embedding Example


```
<tf.Tensor: shape=(16,), dtype=float32, numpy=
array([4.0395218e+01, 3.7746098e+01, 3.0203272e+01, 6.8917831e+01,
       6.1772205e+01, 0.0000000e+00, 4.6411770e+01, 1.2622414e+01,
       3.7707138e+01, 0.0000000e+00, 2.2319033e+01, 6.8590881e+01,
       7.3204234e-02, 5.5983758e+00, 1.0698434e+02, 0.0000000e+00],
      dtype=float32)>
```

LSTM MODEL

LSTM (Long Short-Term Memory) networks are a type of recurrent neural network (RNN) that are particularly effective at learning from sequences of data. They can capture long-term dependencies and mitigate issues like vanishing gradients.

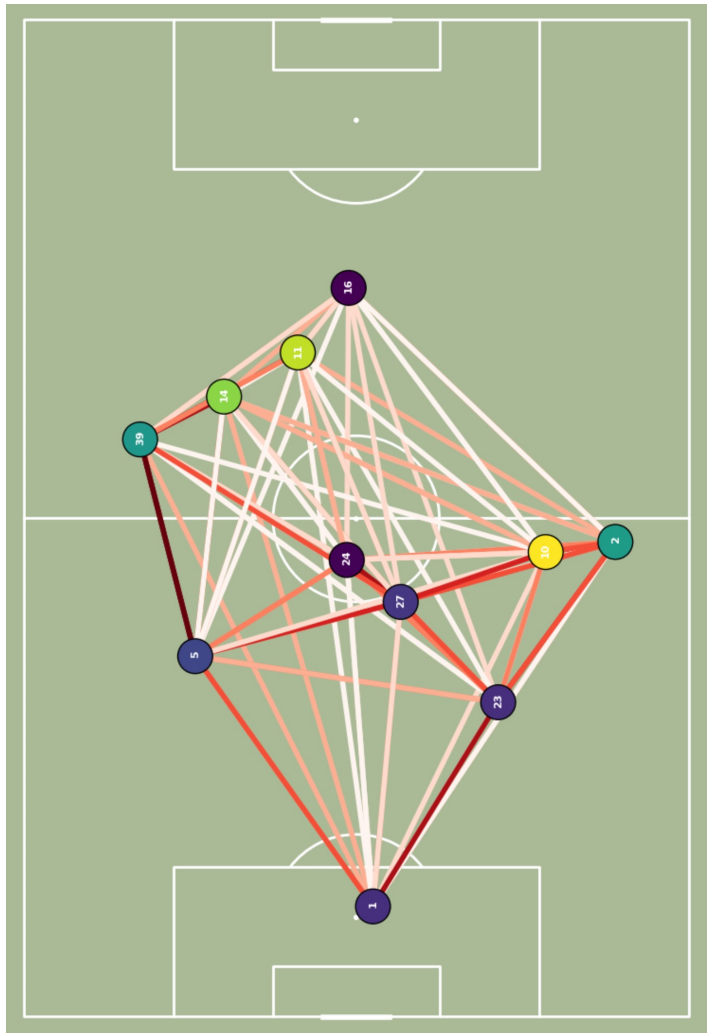
- LSTM layers are used to capture temporal dependencies in sequential data. In this model, multiple LSTM layers are stacked to learn complex patterns from the input sequences.
- The first two LSTM layers use return_sequences=True to ensure they return the entire sequence, allowing the subsequent LSTM layers to process the sequence step-by-step.
- The final LSTM layer only returns the output of the last time step, which is then fed into the Dense layer for the final classification.

```
model = Sequential([
    Reshape((6, 16), input_shape=(6, 16)),
    LSTM(32, return_sequences=True),
    Dropout(0.5),
    LSTM(16, return_sequences=True),
    Dropout(0.2),
    LSTM(8),
    Dense(1, activation='sigmoid')
])

model.compile(optimizer='adam',
              loss='binary_crossentropy',
              metrics=['accuracy'])
```

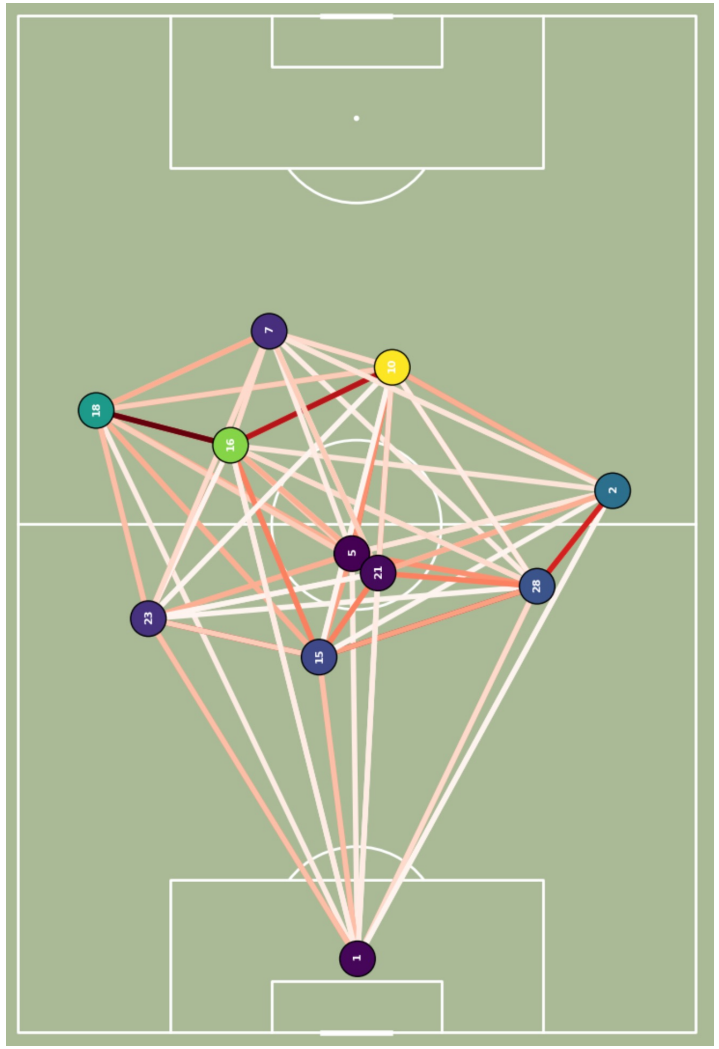
Test Loss: 0.5584357380867004

Test Accuracy: 0.752399206161499

Model 2: Does it make sense ?



Match	Osasuna (0)	Barcelona (2)
Total Passes	358	674
Pass Completion %	83.9%	90.8%
Match Won	NO	YES
Goals Scored	0	2
PassNet Score	0.35017148	0.5710833



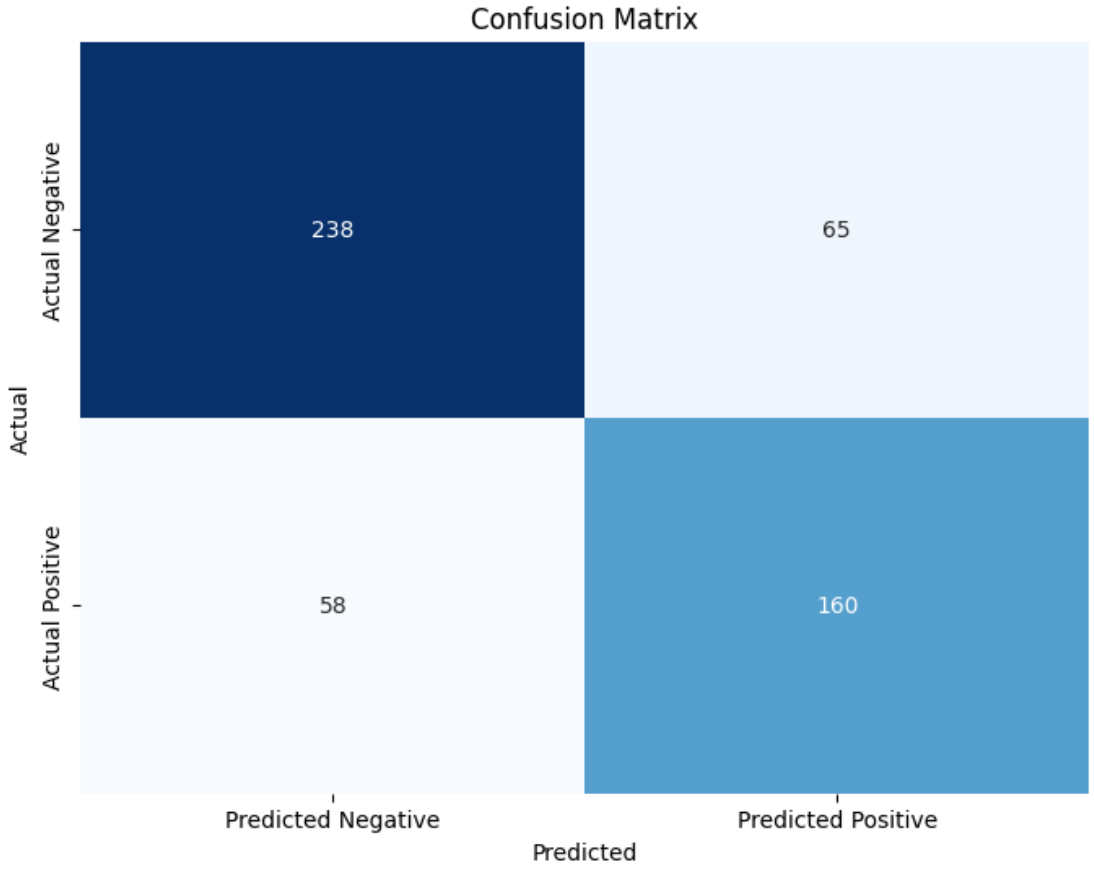
Osasuna Passing Network

Barcelona Passing Network

Evaluation of Model 2

Models	GNN +LSTM
Accuracy	0.7619
Precision	0.77
Recall	0.61

```
Test Loss: 0.5512987971305847
Test Accuracy: 0.7619961500167847
Test Precision: 0.7732558250427246
Test Recall: 0.6100917458534241
Precision: 0.71
Recall: 0.73
```



Evaluation using Graph Theory Metrics

Density: Measures how densely the graph is connected.

Average Degree: Average number of connections per node.

Clustering Coefficient: Measures the tendency of nodes to form clusters.

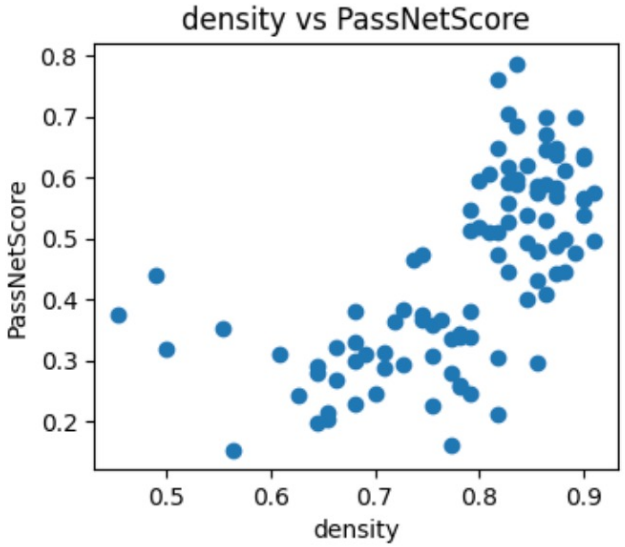
Average Path Length: Average shortest path length between all pairs of nodes.

Betweenness Centrality: Indicates nodes that act as bridges between different parts of the graph.

Eigenvector Centrality: Measures the influence of nodes based on their connections.

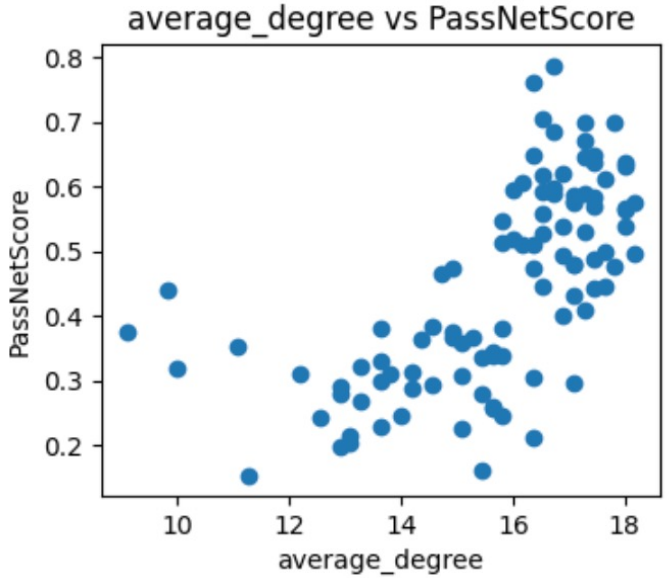
Transitivity: Measures the overall clustering in the graph.

Evaluation Using Graph Theory Metrics



Density: Measures how densely the graph is connected.

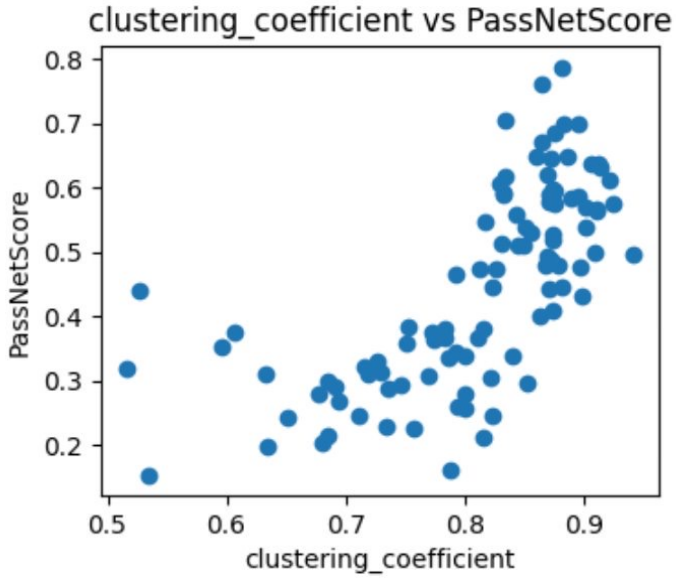
High density indicates a well-connected team with frequent interactions among players.
As we can see from the above plot, PassNet Score increases with an increase in density

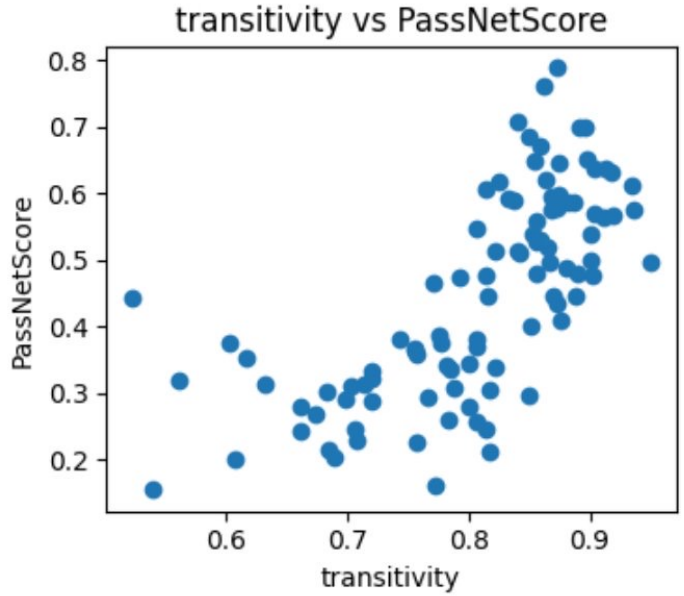


Average Degree: Average number of connections per node.

Indicates the average involvement of players in the passing
network. High involvement of players in a network should increase the PassNet Score which is evident from the plot above

Evaluation Using Graph Theory Metrics

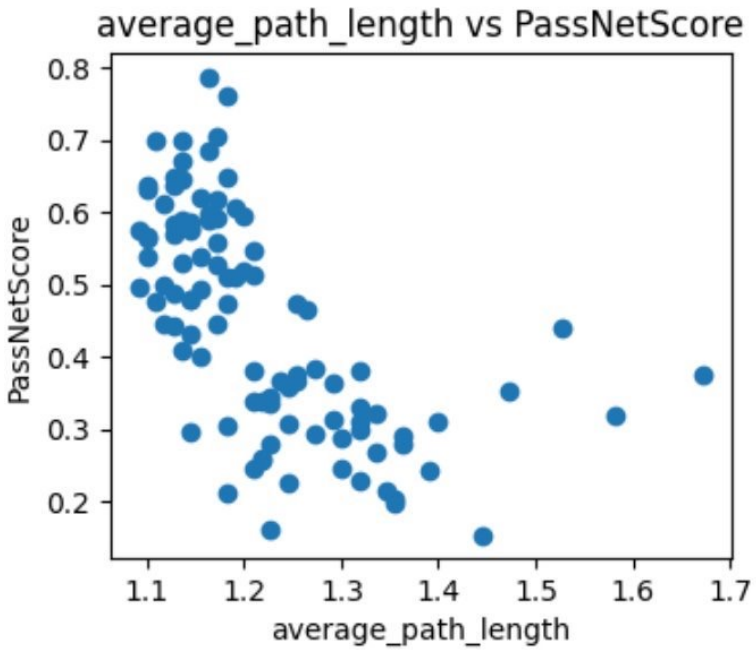




Clustering Coefficient: Measures the tendency of nodes to form clusters.
High clustering indicates that players frequently pass within small groups, which may imply effective short passing strategies. Passes within small group can help build better connections within Passing Networks which can be seen from the plot above

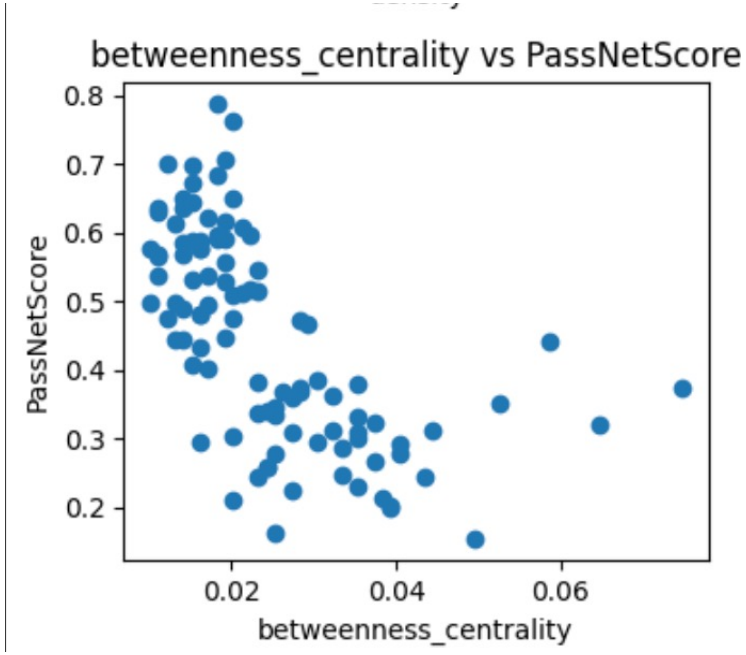
Transitivity: Measures the overall clustering in the graph.
High transitivity indicates strong local connections, which can be crucial for maintaining possession. As we see from the graph above PassNet Score increases with increase in transitivity

Evaluation Using Graph Theory Metrics



Average Path length: Average shortest path length between all pairs of nodes.

Short average path lengths indicate efficient ball movement and connectivity across the team. This also shows how connected a team is and can be evaluated as a decrease in PassNet Score with an increase in average path length

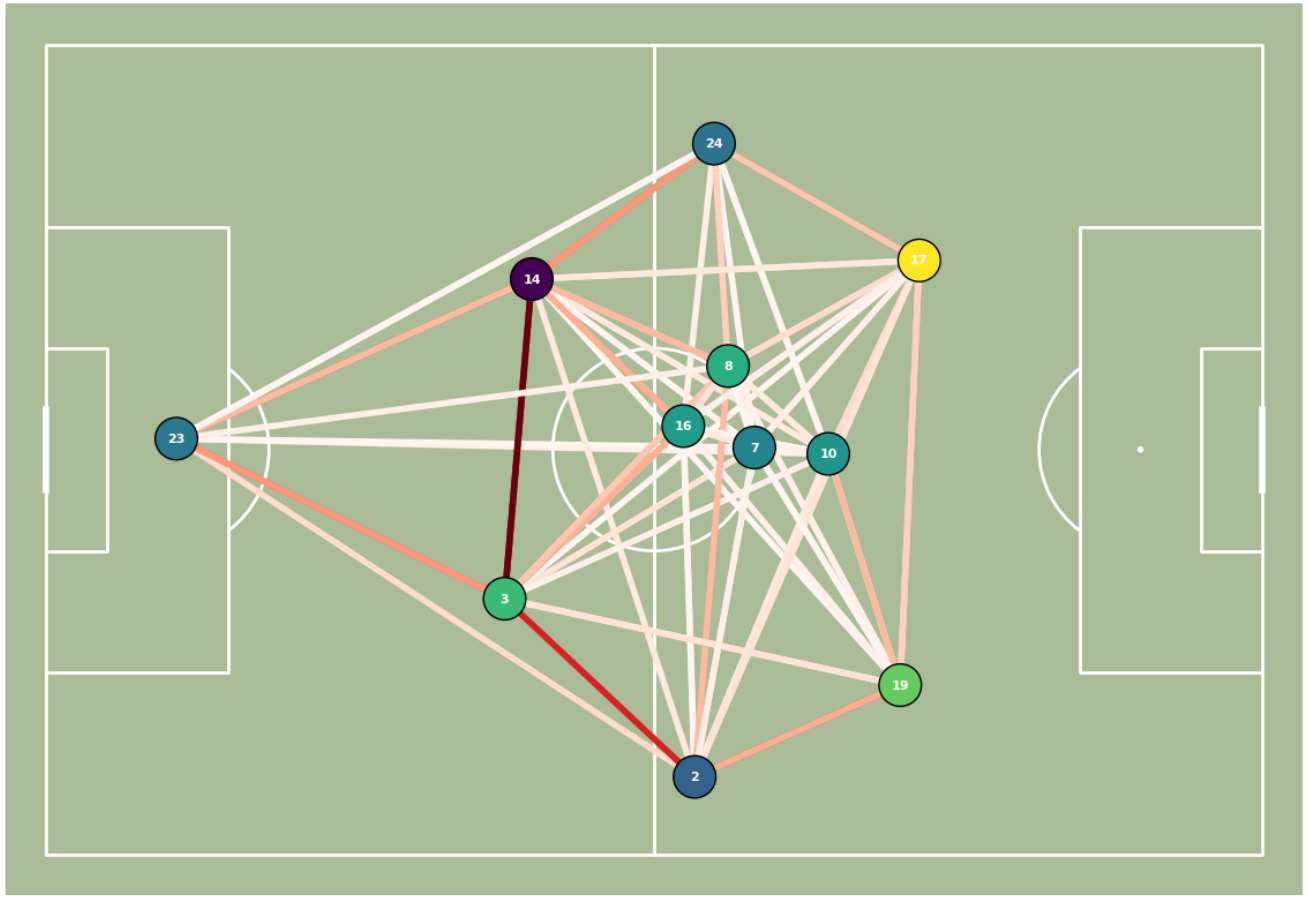


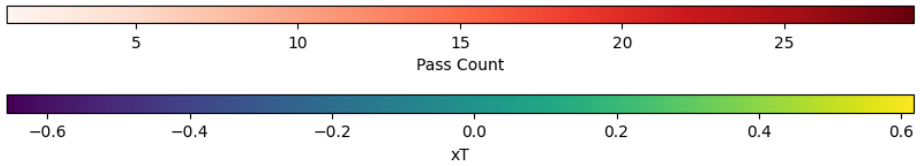
Betweenness Centrality: Indicates nodes that act as bridges between different parts of the graph.
High centralization indicates reliance on a few players for passing, which can be a strength or a vulnerability. In this case we can see that PassNet Score decreases with an increase in betweenness centrality

EURO 2024 FINAL ANALYSIS

PASSNET SCORE

Spain





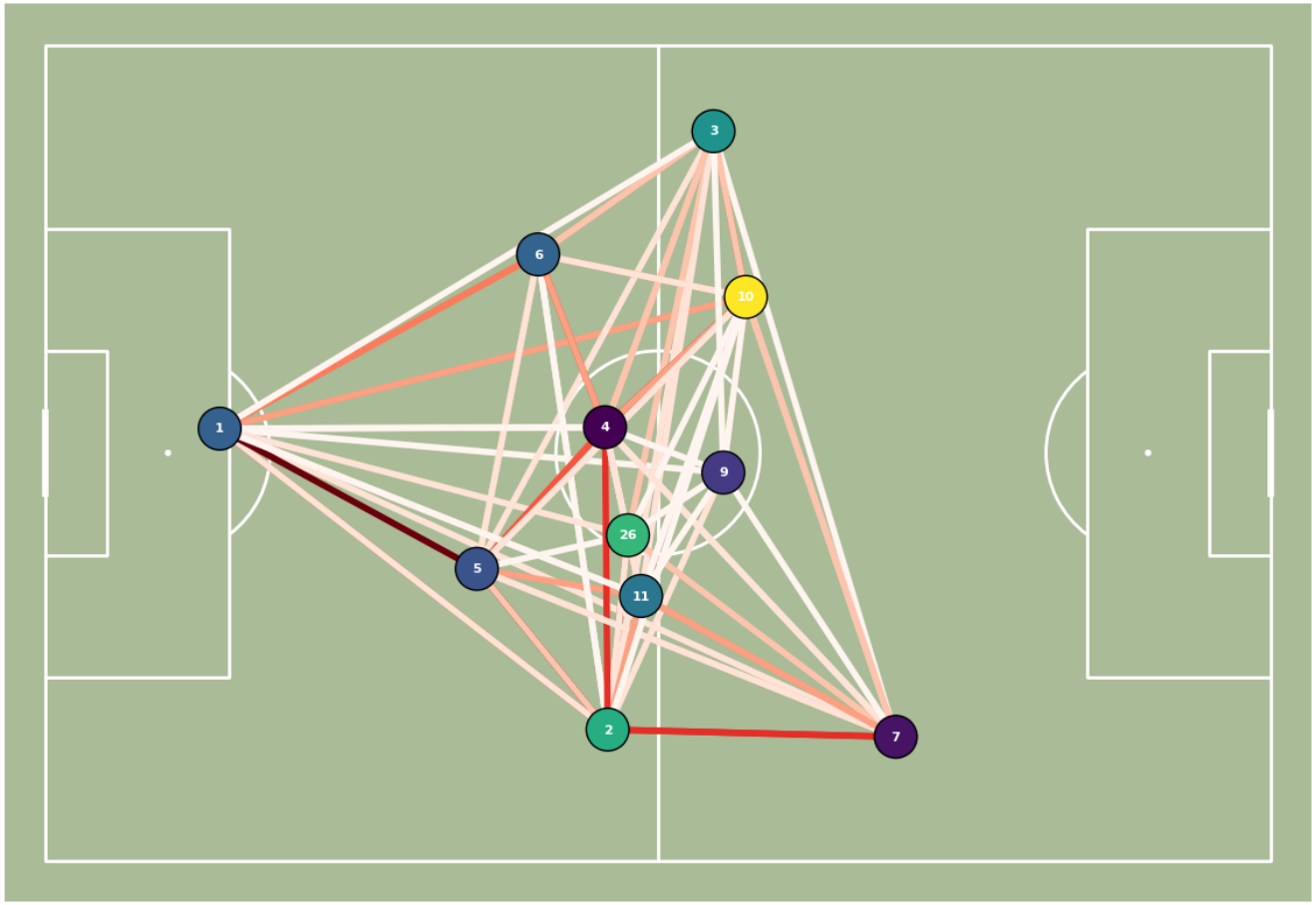
PassNet Score: 0.680513

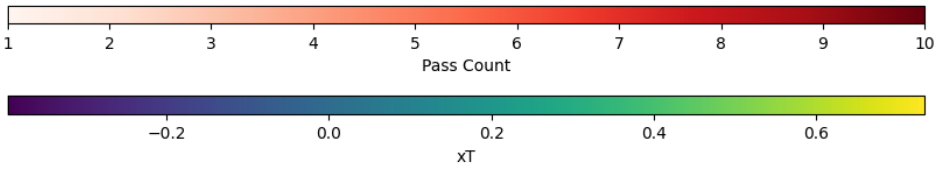
Total Pass	**564**
Pass Completion %	**91.8**
Goals	**2**

Key Points

1. A much better network structure
2. Well distributed xT and Pass counts.
3. Average position is higher up the field

England





PassNet Score: 0.210513

Total Pass	**315**
Pass Completion %	**80.6**
Goals	**1**

Key Points

1. Network is skewed to the right.
2. More players toward lower xT & Pass count gradient.
3. Average position of the players are in the middle of the pitch.

CONCLUSION

FUTURE WORK

1. Collect More Data
2. Predicting Goals in Future Periods
3. Incorporate Additional Metrics
4. Enhance Model Architecture


THANK YOU