Graph-Based vs. Feature-Based Tennis Match Prediction: A Comparative Study

Donato Festa
Università degli Studi di Bari Aldo Moro
Email: d.festa5@studenti.uniba.it

***Abstract*—The prediction of professional tennis match outcomes is a challenging task due to the complex interplay of player skills, dynamic form, and contextual factors. Traditional machine learning approaches rely on hand-crafted features and treat matches as independent events, potentially overlooking the rich relational structure of the sport. This project explores this gap by developing and comparing two distinct modeling paradigms: a feature-based XGBoost classifier and a modern Graph Neural Network (GNN).**

The XGBoost model serves as a strong baseline, trained on a comprehensive set of engineered features including dynamic Elo ratings, head-to-head statistics, and fatigue metrics. In contrast, the GNN model learns powerful player representations (embeddings) directly from a directed *winner-loser* graph, using link prediction to capture latent attributes of skill and reputation from the network structure alone.

To ensure a rigorous comparison, both models were evaluated using an identical temporal cross-validation protocol, repeated over 10 independent runs. The results demonstrate that the GNN model (mean Log-Loss: 0.5045) achieves a statistically significant performance improvement ($p < 0.001$) over the baseline (mean Log-Loss: 0.6337). Qualitative analysis of the learned embeddings via t-SNE further reveals that the GNN successfully captures intuitive real-world structures, such as clustering players by their competitive era. This project demonstrates that modeling the relational structure of player interactions is a highly effective strategy, offering superior predictive performance by uncovering patterns not easily captured by traditional statistical methods.

I. Introduction

The prediction of outcomes in professional sports has long been a captivating challenge for both statisticians and machine learning practitioners. Among these, professional men's tennis stands out as a particularly rich domain for analysis. Its dynamic nature, where match results are determined by a complex interplay of individual skill, physical endurance, psychological state, and environmental factors such as the playing surface, presents a formidable modeling problem. Developing accurate predictive models in this space not only holds commercial value in areas like betting markets but, more importantly, provides deeper insights for coaches, analysts, and fans into the factors that drive player performance and shape the landscape of the sport.

However, traditional modeling approaches often struggle to capture the full complexity of these interactions. Many methods rely on aggregated statistics (e.g., win-loss records, ace percentages) or official ranking systems. While useful, these metrics are often lagging indicators of a player's true current form. More critically, they typically treat each match as an independent, isolated event. This paradigm overlooks a fundamental truth of competitive sports: a player's strength is not an absolute quantity but is inherently **relational**, defined by the context of whom they have defeated and to whom they have lost over time. Pioneering work in sports analytics has sought to address this by creating sophisticated hand-crafted features, such as the **Elo rating system** [1], to better capture a player's dynamic form relative to their opponents. While these feature-based methods, often paired with powerful classifiers like XGBoost, represent a strong state-of-the-art, they still do not explicitly model the complete, higher-order network of interactions that defines the entire competitive ecosystem.

Recent advancements in deep learning on graphs, particularly Graph Neural Networks (GNNs), offer a powerful new paradigm to address this limitation. GNNs are a class of neural networks specifically designed to learn from data structured as graphs, making them ideally suited to model the relational ecosystem of a professional sports league. While GNNs have seen successful application in team sports, for instance in analyzing team formations or player movements [2], their application to a global, historical outcome prediction task in an individual sport like tennis remains a less explored area. By representing players as nodes and matches as directed edges, a GNN can learn latent vector representations, or **embeddings**, that encode a player's skill, style, and *reputation* not from predefined statistics, but directly from the structure of the graph itself.

This project aims to fill this gap by conducting a rigorous, comparative investigation into the efficacy of a GNN-based approach for tennis match prediction. We hypothesize that a model capable of learning from the entire relational network can uncover predictive signals that are inaccessible to traditional, instance-based models.

Our main contributions are as follows:

- We develop a robust baseline model using **XGBoost**, trained on a rich set of dynamically calculated features, including Elo ratings, head-to-head statistics, and fatigue proxies, to serve as a strong benchmark.
- We propose a **Graph Neural Network** model for link prediction, based on a GraphSAGE architecture, which learns powerful player embeddings directly from a directed graph of historical match outcomes.
- We conduct a rigorous comparative evaluation using an identical **temporal cross-validation** protocol for both

models, ensuring a fair and methodologically sound comparison.
- We demonstrate that the GNN model achieves a **statistically significant improvement** in predictive performance over the strong feature-engineered baseline.
- We provide a **qualitative analysis** of the learned player embeddings, showing that they capture meaningful and intuitive structures, such as clusters of players from the same competitive era, thereby offering insights beyond simple prediction accuracy.

II. Related Work

The task of predicting sporting outcomes has been approached from two primary angles. This section reviews both the traditional feature-based models and the more recent graph-based approaches to contextualize our work.

A. Feature-Based and Statistical Models

The most established approach to sports prediction relies on meticulous feature engineering. In tennis, this involves creating a feature vector for each match based on player rankings, recent performance, and surface-specific statistics. A cornerstone of this paradigm is the **Elo rating system**, originally developed for chess [1], which provides a dynamic measure of a player's strength by updating their score after each match based on the outcome and the opponent's rating. These features are typically fed into powerful machine learning classifiers, with Gradient Boosted Trees like **XGBoost** [3] being a state-of-the-art choice for their high performance on tabular data.

While highly effective, these models fundamentally operate on a per-match basis. They excel at capturing a player's current form but treat each contest as an isolated event defined by explicit features. They do not inherently model the broader network of relationships or the *reputational* strength a player holds within the entire competitive ecosystem, which is shaped by higher-order interactions.

B. Graph-Based Approaches in Sports Analytics

A more recent paradigm involves representing the domain as a graph, where entities (players or teams) are nodes and their interactions (matches) are edges. This structure allows for the application of **Graph Neural Networks (GNNs)**, which learn node representations (embeddings) that capture complex relational dependencies by aggregating information from the graph structure itself.

This methodology has shown significant promise in team sports for tasks like analyzing formations or predicting player movements [2]. However, its application for a global, historical outcome prediction task in an individual sport like tennis remains a less-explored area. Most existing graph-based analyses in tennis focus on fine-grained, intra-match events rather than modeling the entire player-vs-player network to learn predictive representations. Our project directly addresses this gap by framing the problem as **link prediction** on a global *who-beat-whom* graph, investigating whether learning from this relational structure provides a predictive edge over traditional feature-based methods.

III. Dataset and Preprocessing

A. Data Source and Scope

The foundation of this study is the comprehensive match-by-match dataset for the men's ATP tour, compiled and maintained by Jeff Sackmann[1]. This publicly available repository provides detailed statistics for all ATP tour level matches. To ensure our models are trained on modern playing styles and a sufficient volume of data, we restricted our analysis to all singles matches from the year 2000 to the end of the 2023 season. This scope yields a rich dataset of over 70,000 matches. Each record contains granular information, including tournament details, match context (e.g., surface, round), player attributes (e.g., rank, age), and in-match statistics (e.g., aces, break points), providing a robust basis for both feature-based and graph-based modeling.

B. Data Preprocessing and Temporal Integrity

A series of critical preprocessing steps were applied to ensure data quality and, most importantly, temporal integrity. The raw data, initially distributed across multiple CSV files (one for each year), was concatenated into a single DataFrame. Key columns, such as the match date, were converted to the appropriate 'datetime' format, while all statistical columns were cast to numeric types, with any conversion errors being handled as 'NaN' values.

The most crucial step was to rigorously sort the entire dataset chronologically by match date and an internal match number. This temporal ordering is fundamental to our methodology, as it ensures that all historical and time-dependent features (such as Elo ratings or winning streaks) are calculated without data leakage from the future. This guarantees that our evaluation protocol, which relies on a temporal split, is valid.

C. Exploratory Data Analysis (EDA)

An initial EDA was conducted to understand the dataset's underlying properties and to validate our modeling assumptions. Figure 1 illustrates one of the most significant findings: a clear and stark difference between the rank distributions of winners and losers. Winners are overwhelmingly concentrated in the top-tier of the rankings (lower rank number), while losers' ranks are more broadly distributed. This strong negative correlation between rank and winning confirmed that rank-based features, and by extension more dynamic rating systems like Elo, would be highly predictive and must form the core of any competitive baseline model.

Furthermore, the analysis confirmed that the playing surface is a critical contextual factor. A majority of matches are played on Hard courts (approximately 50%), followed by Clay (35%) and Grass (15%), with a small fraction on Carpet, which actually isn't used anymore. This distribution validated our decision to include surface type as a key categorical feature in our baseline model.

[1] https://github.com/JeffSackmann/tennis_atp

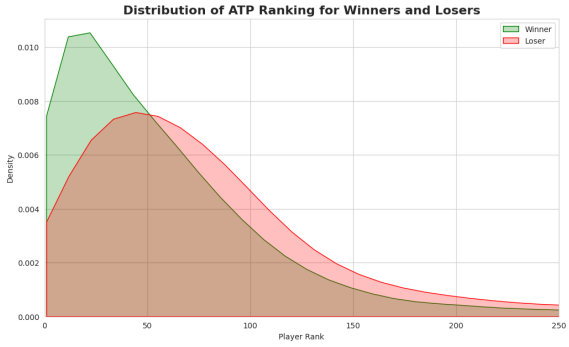

Figure 1: Probability density distribution of player rankings for match winners and losers. A lower (better) rank is highly correlated with winning.

IV. Methodology

This project employs a direct comparative approach, evaluating a traditional feature-based model against a modern graph-based model. This section details the data and feature preparation, the architecture of both models, and the robust evaluation protocol designed to ensure a fair comparison.

A. Baseline Model: XGBoost with Engineered Features

The baseline model is designed to be a strong competitor, representing a state-of-the-art approach based on traditional feature engineering. We use XGBoost [3], a powerful and highly effective gradient boosting algorithm, as our classifier.

1) Feature Engineering: The performance of the XGBoost model relies on a rich set of features calculated for each match. These features are computed iteratively over the chronologically sorted dataset. For each match, we create a vector of **differential features**, which directly compare the two competing players, as well as contextual features. The most critical engineered feature is the **Elo rating** [1]. We calculate the expected win probability E_1 for a player as:

$$E_1 = \frac{1}{1 + 10^{(R_2 - R_1)/400}} \tag{1}$$

where R_1 and R_2 are the pre-match Elo ratings. The ratings are updated after each match based on the actual outcome, providing a more responsive measure of current form than official rankings. Other key differential features include differences in **ATP rank**, **head-to-head** records, current **winning streaks**, **age**, and a **fatigue** proxy based on minutes played in the last 7 days. Contextual features include one-hot encoded vectors for the court surface (Hard, Clay, Grass, Carpet).

2) Data Structuring for Modeling: To prevent positional bias (i.e., the model learning to favor the player listed first), the dataset was structured such that the assignment of *Player 1* and *Player 2* for each match was randomized. For each match, two rows were conceptually created: one where the actual winner is Player 1 (target=1) and one where the winner is Player 2 (target=0). Approximately 50% of matches were randomly selected to use the first representation, and the other 50% used the second. This forces the model to learn from the feature differences rather than their arbitrary position in the input vector.

B. Graph-Based Model: GNN for Link Prediction

Our primary model recasts the task from a standard classification problem to a **link prediction** problem on a directed graph. This approach aims to learn player representations (embeddings) not from their individual statistics, but from their relational position within the entire network of matches.

1) Graph Construction: We construct a single, global graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ from the training data of each cross-validation fold. In this graph:

- $\mathcal{V}$ is the set of nodes, where each node represents a unique player present in that training period. A mapping is created from player ID to a unique node index.
- $\mathcal{E}$ is the set of directed edges. A directed edge $(u, v) \in \mathcal{E}$ exists if player u defeated player v.

This structure captures the complex web of historical dominance and upsets, forming the sole source of information for the GNN model.

2) Model Architecture and Training: We implement a 'GNNLinkPredictor' model using PyTorch Geometric [4], which consists of an encoder-decoder architecture.

- **Encoder**: A two-layer **GraphSAGE** network [5] generates a final embedding $\mathbf{z}_u \in \mathbb{R}^d$ for each player u. The model learns these embeddings from scratch, starting with an initial random vector for each player from a 'torch.nn.Embedding' layer and refining it by aggregating information from graph neighbors.
- **Decoder**: A dot product function serves as the decoder. Given embeddings $\mathbf{z}_u$ and $\mathbf{z}_v$, it calculates a logit score $s_{uv} = \mathbf{z}_u \cdot \mathbf{z}_v$ for the potential directed edge (u, v). This score is passed through a sigmoid function to produce the win probability.

The model is trained end-to-end to minimize a binary cross-entropy loss. For each positive edge (a real match 'winner $->$ loser'), a corresponding **negative sample** (a non-existent edge, e.g., 'winner $->$ random_player') is generated via uniform negative sampling. This forces the model to learn embeddings that produce a high score for true links and a low score for false ones, implicitly learning the directional *winner-beats-loser* relationship. An L2 regularization term is added to the loss to prevent overfitting on the embedding weights. The model is trained using a full-batch approach; in each epoch, the encoder processes the entire training graph to generate embeddings, and the loss is computed over all positive and newly-sampled negative edges in a single pass.

C. Evaluation Protocol

To ensure a robust and fair comparison, both models were evaluated using an identical, rigorous protocol.

- **Temporal Cross-Validation:** We employed a *TimeSeriesSplit* with 5 folds. This method is critical for time-series data, as it ensures that the model is always trained

on past data to predict future, unseen matches, mimicking a real-world scenario and preventing data leakage.

- **Multiple Runs & Hyperparameter Search:** The entire 5-fold cross-validation process was repeated **10 times** for each model to yield a stable distribution of performance scores.
  - For the **XGBoost baseline**, within each fold of each run, a 'RandomizedSearchCV' was performed to find the best hyperparameters from a predefined distribution space. Each of the 10 main runs thus represents an independent end-to-end optimization and evaluation cycle.
  - For the **GNN**, each of the 10 runs was trained and evaluated using a different, pre-defined set of hyperparameters to explore a variety of model configurations.
- **Primary Metric:** Our primary evaluation metric is **Log-Loss**, chosen for its ability to measure the accuracy of predicted probabilities. It heavily penalizes overconfident and incorrect predictions, making it a stringent measure of model performance.

The final comparison is based on a **paired t-test** on the 10 out-of-fold Log-Loss scores generated for each model.

V. Results and Analysis

This section presents the results of our experimental evaluation. We first provide a quantitative comparison of the predictive performance of the XGBoost baseline and the GNN model, including a statistical test of significance. We then delve into a qualitative analysis of the GNN's learned embeddings to interpret the relational structures the model has captured.

A. Quantitative Performance Comparison

The performance of both models was evaluated using the robust protocol described in Section IV. For each of the 10 independent runs, we computed the overall out-of-fold Log-Loss, our primary metric for evaluating the accuracy of probabilistic predictions. Table I summarizes the descriptive statistics of these scores for both models.

The results clearly indicate the superiority of the GNN model. On average, the GNN achieves a mean Log-Loss of **0.5045**, a substantial improvement over the strong XGBoost baseline's mean of 0.6337. The GNN not only performs better on average but also demonstrates robustness, with its worst-performing run (Log-Loss: 0.5267) still significantly outperforming the best run of the baseline (Log-Loss: 0.6223).

To verify that this large performance gap is not a product of random chance, we performed a paired t-test on the 10 Log-Loss scores from each model. The test yielded a t-statistic of **36.66** and a p-value of approximately $\mathbf{4.1 \times 10^{-11}}$. As this p-value is far below the standard significance level of $\alpha = 0.05$, we can confidently reject the null hypothesis and conclude that the GNN model's superior performance is **statistically significant**.

Table I: Summary of model performance over 10 evaluation runs. The GNN model demonstrates a substantially lower (better) mean Log-Loss.

Metric	XGBoost Baseline	GNN Model
Mean Log-Loss	0.6337	**0.5045**
Std. Dev. Log-Loss	0.0081	0.0126
Min Log-Loss (Best)	0.6223	0.4923
Max Log-Loss (Worst)	0.6480	0.5267

B. Qualitative Analysis of Player Embeddings

Beyond quantitative metrics, it is crucial to understand *what* the GNN has learned. To do this, we analyzed the 64-dimensional player embeddings generated by the best-performing GNN model run. We used the t-SNE algorithm to project these high-dimensional vectors into a 2D space for visualization, with the results shown in Figure 2.

The visualization reveals that the GNN, without any prior knowledge of the players, has learned a meaningful organization of the player ecosystem. Distinct clusters and structures have emerged automatically from the graph data. Most notably, a prominent and dense cluster, highlighted in Figure 2, contains the dominant figures of modern tennis's *Golden Era*: Roger Federer, Rafael Nadal, Novak Djokovic, and Andy Murray. The model has correctly identified these players as relationally similar.

Interestingly, this cluster also includes players like Andreas Seppi and Gaël Monfils. While not typically ranked alongside the *Big 4*, their inclusion provides a deeper insight into the model's logic. These players are characterized by exceptional career longevity and, as a result, are highly-connected *hub* nodes in the graph who frequently played against the same elite opponents. For instance, our analysis shows that Seppi played 65 matches against a small group of elite players from that era. This suggests that the GNN has learned to represent not just raw skill, but also a player's **role and context** within their competitive era. This ability to capture deep, latent relational features is likely the primary reason for its superior predictive performance, as it moves beyond simple win/loss statistics to model a player's *reputation* and connectivity.

VI. Conclusion and Future Work

This project conducted a rigorous comparative study between a traditional, feature-based XGBoost model and a modern, graph-based GNN for predicting outcomes in professional men's tennis. Our findings confirm the primary hypothesis: modeling the relational network of players provides a more powerful predictive signal than relying on instance-based statistical features alone. The GNN model achieved a mean Log-Loss of **0.5045**, a statistically significant improvement over the strong baseline's 0.6337 ($p < 0.001$). Our qualitative analysis further revealed that the GNN's learned embeddings capture intuitive, real-world structures, such as clustering players by competitive era, demonstrating its ability to learn a player's *reputation* and context from the graph structure.

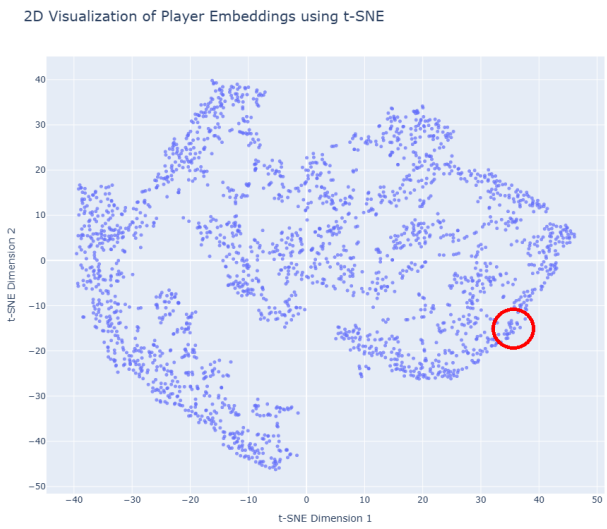

Figure 2: 2D t-SNE visualization of the learned player embeddings. The model has automatically organized players into distinct clusters. The highlighted region shows the *Golden Era* cluster, centered around Federer, Nadal, Djokovic, and Murray.

However, our analysis also highlighted a key limitation. While powerful, the simple dot-product decoder used in our GNN tends to produce over-confident predictions (probabilities near 1.0 or 0.0) for historical matchups with a clear dominant player. This suggests that while the model effectively learns long-term dominance patterns, it may not fully capture the inherent stochasticity of any single match.

These findings open several exciting avenues for future work. A direct next step to address the over-confidence issue would be to explore more sophisticated **decoder architectures**. Replacing the dot product with a Multi-Layer Perceptron (MLP) could allow the model to learn more complex, non-linear interactions between player embeddings, potentially leading to better-calibrated probabilities. Our preliminary experiments suggested this path requires a more advanced hyperparameter optimization strategy.

Furthermore, the current GNN learns from a simple, unweighted graph. Its performance could be significantly enhanced by creating a **heterogeneous**, feature-rich graph. This would involve adding static attributes (e.g., height, handedness) as initial **node features** and contextual information (e.g., court surface, tournament importance) as **edge features**. This would allow the GNN to learn context-dependent relationships (e.g., how player A's style fares against player B's specifically on clay).

A powerful and pragmatic direction would be to create a **hybrid model** that combines the strengths of both paradigms investigated in this project. This approach would involve using the GNN's learned embeddings as rich, relational features within the high-performing XGBoost framework, alongside the dynamic features like Elo and fatigue. Such a model could leverage both the long-term, structural knowledge captured by the GNN and the short-term, up-to-the-minute form captured by traditional metrics.

Finally, the most advanced extension would be to move from a static graph to a dynamic one. Implementing a **Temporal Graph Network (TGN)** would represent a paradigm shift, allowing player embeddings to evolve continuously over time. This would enable the model to capture a player's changing form and momentum directly within the graph structure, offering the most holistic and powerful approach to the problem.

Code and Data Availability

The source code, experimental results, and analysis notebook for this project are publicly available in my GitHub repository: https://github.com/DonatoFe11/Tennis-Match-Predictor. The dataset used is from the public repository maintained by Jeff Sackmann.

Appendix A
Hyperparameter Search Spaces

This section details the hyperparameter search spaces used for both the XGBoost baseline and the GNN model evaluations.

A. XGBoost Baseline

The baseline model was optimized using 'RandomizedSearchCV' over the following parameter distributions for each of the 10 evaluation runs.

Listing 1: Parameter distribution for XGBoost baseline.

```
param_dist = {
    'learning_rate': uniform(0.01, 0.15),
    'max_depth': randint(3, 10),
    'n_estimators': randint(500, 2500),
    'subsample': uniform(0.7, 0.3),
    'colsample_bytree': uniform(0.7, 0.3),
    'gamma': uniform(0, 0.5)
}
```

B. GNN Model

For the GNN, each of the 10 evaluation runs used one of the following predefined hyperparameter configurations from our experimental grid.

Table II: Hyperparameter grid for the 10 GNN evaluation runs.

Run	LR	Emb Dim	Hidden Ch	Out	Dropout	Epochs
1	0.005	128	128	64	0.5	200
2	0.010	128	256	128	0.4	150
3	0.001	128	128	64	0.3	250
4	0.005	64	256	64	0.5	200
5	0.010	64	128	32	0.4	150
6	0.001	256	256	128	0.3	250
7	0.008	128	128	64	0.5	200
8	0.003	256	256	128	0.4	150
9	0.005	64	64	32	0.3	250
10	0.010	128	128	64	0.5	200

References

[1] A. E. Elo, *The Rating of Chessplayers, Past and Present*. New York: Arco, 1978.

[2] T. T. Drexler, "Sports analytics with graph neural networks and graph convolutional networks," *Preprints.org*, 2024, version 1, Posted: 1 October 2024.

[3] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '16. New York, NY, USA: ACM, 2016, pp. 785–794.

[4] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[5] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems 30*, ser. NIPS '17, 2017, pp. 1024–1034.