Evaluating Player Actions in Professional Counter Strike using Temporal Heterogeneous Graph Neural Networks

Other Sports
Paper ID: 20251422

1. Introduction

In recent years, electronic sports (esports) have gained popularity, extending the existing landscape of the sports industry. Counter Strike 2 (CS2), a first-person shooter team game, stands as one of the most prominent esports titles in 2024. In this esport, two teams face off within a match, taking turns as attackers (Terrorists - Ts) and defenders (Counter Terrorists - CTs). A match consists of 2-minute rounds where Ts must plant a bomb at one of two bomb sites, while CTs must prevent it or defuse the bomb. The first team to win 13 rounds wins the match. With tournaments organized in front of large audiences and professional teams competing for substantial prize pools, the stakes of the professional scene are high. Despite these facts and the abundance of available data, only a few artificial intelligence-driven solutions have been explored so far regarding individual and team performance enhancement [1, 2, 3, 4], and it has not yet gained much popularity in practical use.

This paper proposes a framework capable of evaluating player actions using graph neural networks (GNNs). This involves a reliable and consistent data-transforming process capable of creating heterogeneous graph datasets by parsing publicly available professional match replay files at a desired framerate. These heterogeneous graphs incorporate the players and the unique layout of the map the game is held on using different node types and connections. Extending this method, graphs are concatenated across time, creating discrete-time dynamic graphs. Leveraging the data, temporal heterogeneous GNNs (THGNNs) are trained to predict whether the attacker or the defender team would win the examined round. With models able to accurately predict round-winning chances at a frame-by-frame level, fluctuations in these probabilities are then examined within single rounds. The events prior to the probability changes are collected, and we use Shapley values to measure the contributions of the events for changes in winning chances. By associating players with the events, the impact of the individuals can be examined.

The proposed framework, with GNN models exceeding the 76% accuracy mark, enables detailed analysis of key in-game events, such as opening kills and clutch plays, by measuring their impact on win probability. An illustration is visible on Figure 1.1, highlighting the impact of an opening elimination. The framework also excels at highlighting seemingly unimportant events that may not be immediately obvious, but still influence the team winning chances. This provides teams with valuable information about which actions are most impactful, regardless of their perceived significance. Additionally, the framework provides insights into how identical events, such as an opening kill, can have varying impacts depending on the context, such as weaponry or map positioning. This can help teams better understand the specific factors that amplify or reduce the effectiveness of similar actions across different scenarios.

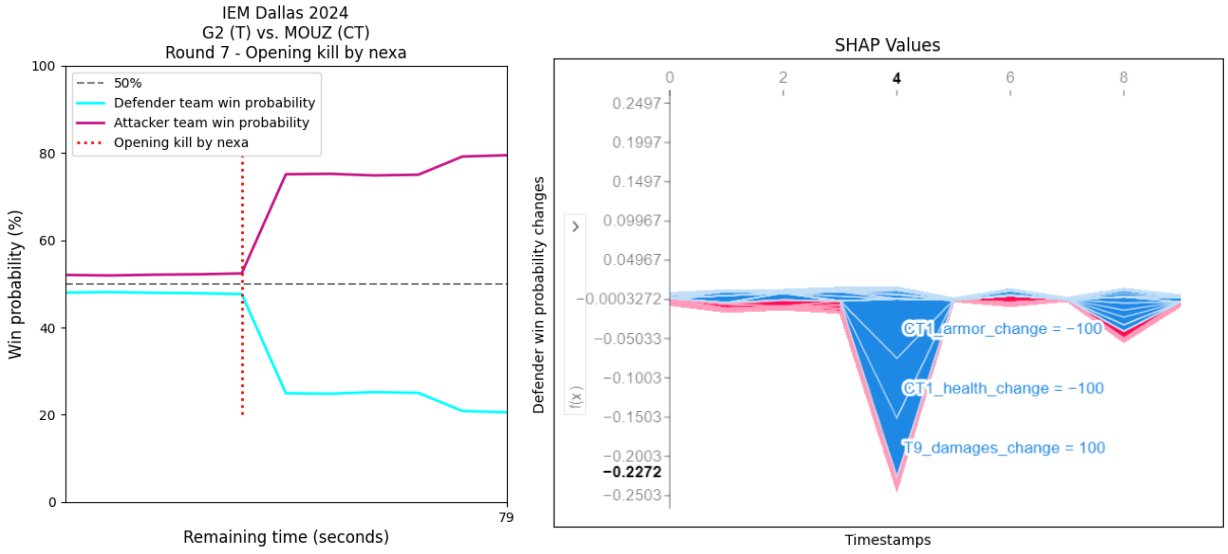


Figure 1.1: Analysis of an opening kill performed by the attacker called nexa in a match between teams G2 Esports and MOUZ. (a) The win probabilities predicted for the teams at the time of the event, with the opening kill timestamp highlighted. (b) The corresponding Shapley values highlight the most important factors regarding the event: the defender (labeled CT1) losing all their health and armor, and nexa (labeled as T9) increasing his damage stat, resulting in a 22% win probability decrease for the defenders.

Our framework has the potential to revolutionize individual player performance analysis in the Counter Strike esports industry by providing a detailed, frame-by-frame assessment of player impact. This opens the possibility for detailed player evaluation methods as well as more profound ways of strategic planning, providing organizations with a competitive advantage. This paper is organized as follows. Section 2 outlines the process of creating the graph dataset. Section 3 describes our approach to training and optimizing the GNN model. Section 4 focuses on explanatory analysis and player evaluation. Section 5 presents various use cases for our framework in application. Section 6 reviews the related work in the field. Section 7 concludes the paper.

2. Dataset creation

An advantage of esports, compared to traditional sports, is that it takes place in a virtual environment. This factor makes the collection of tracking data more accessible. In the case of CS2, demo files—exact copies of actual matches—are publicly available for download via a platform called HLTV[1]. For this study, we collected all professional match replays from S-tier tournaments held between the release of CS2 in October 2023 and September 2024, focusing the analysis on matches played on the map called Inferno. This yielded a total of 114 matches.

To create the initial tracking dataset, the *awpy* package [1] is utilized, which efficiently extracts all essential data from the replay files. The matches were parsed at a rate of four snapshots per second. The resulting tracking data contains all state-describing features of the players, including their

[1] HLTV website: https://www.hltv.org/

positions, velocities, weapons, inventories and health points. It also includes grenade trajectories, among other key attributes. Previous work by Xenopoulos et al. (2020) demonstrated that this level of tracking data in tabular format can be utilized for uncovering patterns within the game. However, Counter-Strike presents a unique challenge compared to traditional sports or other esports due to the variability in map layouts. Each map features its own distinctive structure and design, requiring strategies and tactics that are intricately tied to its specific playfield (e.g., Inferno). Players must adapt their gameplay to the unique features of the environment, employing map-specific tactics. To address this complexity, we decided to represent the game state using graphs, a format well-suited for capturing both the spatial layout of maps and the dynamic interactions between players and their environment.

2.1. Modeling the map layout

The construction of the map graph was assisted by a former professional player, whose expertise played a crucial role in the accurate representation of the map. The task involved determining the placement of nodes V and adding edges E between them to form the $G_{map} = (V, E)$ map layout graph. A key consideration was balancing node placement to avoid redundancy from excessive density while ensuring the map's layout was accurately captured. We concluded that each node $v_i \in V$ should represent an area approximately the size of a smoke grenade's coverage. Smoke grenades are crucial for strategic play, as they obscure key positions or pathways intentionally aligned with their size by map designers. This principle not only highlights important smoke points but also serves as an efficient unit for covering the map. The nodes were placed on the map accordingly. Each stored state-describing properties in its feature vector, such as X, Y and Z coordinates and boolean flags indicating whether it is on fire or smoked. To complete the graph, edges were added to represent pathways between the vertices. An undirected edge $e_{ij} \in E$ connects nodes v_i and v_j if player movement is possible between them in both directions. To further enrich the environment graph, additional nodes were placed on common contact positions on the map to highlight engagement areas. These contact spots were identified using density-based clustering algorithms applied to player elimination data (frequent kill and death locations), with additional corrections and validation provided by the former professional player.

2.2. Dynamic graph snapshots

The initial step to construct the dynamic graph dataset is the creation of static graphs representing game-states. These snapshots were built upon the predefined map graph, serving as the structural foundation for representing the game state. Using the tracking data, each snapshot was generated by adding the players as nodes to the map graph, positioned at their respective X, Y, Z coordinates to reflect their in-game locations. These *player* nodes serve a fundamentally different purpose than the *map* nodes, thus requiring significantly more features to be stored. As such, they were introduced as a new node type. This resulted in the graph being heterogeneous, comprising *map* type nodes with *map-to-map* edges representing the map layout, and *player* type nodes representing players. The latter node type contains a rich set of features, including state-describing attributes (e.g., X, Y, Z coordinates, velocity, view direction, weapons carried and equipped, health points), match statistics (e.g., damage dealt, successful eliminations, deaths), and general performance metrics from the

previous competitive year (e.g., HLTV Rating 2.1 [5], KAST[2] or Impact[3]). The overall feature count for this node type exceeds 160. Each *player* node was connected to the nearest *map* node with a *player-to-map* edge type, and a self-loop edge labeled *player-is-player* was also added, with its role to be clarified later.

Figure 2.1: An example graph from the created dataset. The game state is from a quarterfinal match between teams Cloud9 and Heroic from the BLAST Premier Fall Final 2023 tournament.

Graph-level features independent of individual nodes were also integrated into the graphs. These included general attributes, such as the remaining time, whether the bomb is planted, and if so, its coordinates and location. Aggregated team-level features were also included, for instance, the number of players alive, total team health, and the team equipment value. Additionally, the graph-level vector also stored the output variable: a boolean flag indicating whether the defending team won the round. An example graph snapshot is visualized in Figure 2.1.

To account for the temporal dynamics of this esport, Discrete-Time Dynamic Graphs (DTDGs) were created by concatenating 20 consecutive snapshots along the temporal axis to form 5-second time windows. Each round is segmented into such intervals. To enhance data diversity, additional intervals were created with a 2.5-second temporal offset. This preprocessing pipeline was applied to

[2] KAST (Kills-Assists-Survives-Traded) metric explanation from HLTV: percentage of rounds in which the player either had a kill, assist, survived or was traded.

[3] Impact metric explanation from HLTV: Measures the impact made from multikills, opening kills and clutches.

all rounds across the 114 matches, yielding approximately 90.000 DTDGs, which formed the foundation for model training.

3. Predicting the round-winner team

To learn patterns from the dynamic heterogeneous graph dataset, we utilized THGNNs. The task of predicting the round winner is framed as a binary graph classification problem. Importantly, this classification task involves nearly balanced classes, as on the map Inferno, defenders (CTs) win approximately 50.5% of the rounds, while attackers (Ts) win 49.5% on big events according to HLTV. A probability prediction is generated for each of the 20 snapshots in the input dynamic graph provided to the model. Section 3.1 outlines the architectural design of the trained neural networks. Following this, Section 3.2 highlights the convolutional strategy detailing how the spatial dynamics of the graphs are captured. Section 3.3 presents the performance of the best-performing model and analyses its results from various perspectives.

3.1. Model architecture

THGNNs are powerful tools for learning spatio-temporal patterns in graphs, offering a wide variety of convolutional possibilities and solutions for modeling dynamic graph data. In this paper, we propose the most successful strategy among the tested approaches. A dynamic graph, serving as the input to the THGNN, is processed through the network's architecture, which systematically extracts and learns spatial and temporal patterns. The architecture of the proposed network is composed of the following main parts:

1. The first step involves **spatial convolutions**. In this step, the information stored in the nodes is propagated to their neighboring vertices through multiple convolutions, creating embeddings for each node. This operation is completed within each static graph that constitutes the dynamic graph, independently. The convolution strategy is described in detail in Section 3.2.

2. The second step focuses on **learning temporal patterns**. Once the spatial convolutions are applied and the node embeddings have been created, the GraphGRU [7] operator is utilized to capture the temporal dependencies and trends across the 20 static graphs that constitute a dynamic graph. The GraphGRU updates the node embeddings accordingly to reflect these temporal patterns. The network includes two GraphGRU layers to ensure separate learning for the two node types.

3. The last step involves the flattening of the static graphs into a batch of 20 vectors and passing them through **dense layers** in the network to produce the final predictions.

Figure 3.1 visualizes the described network architecture.

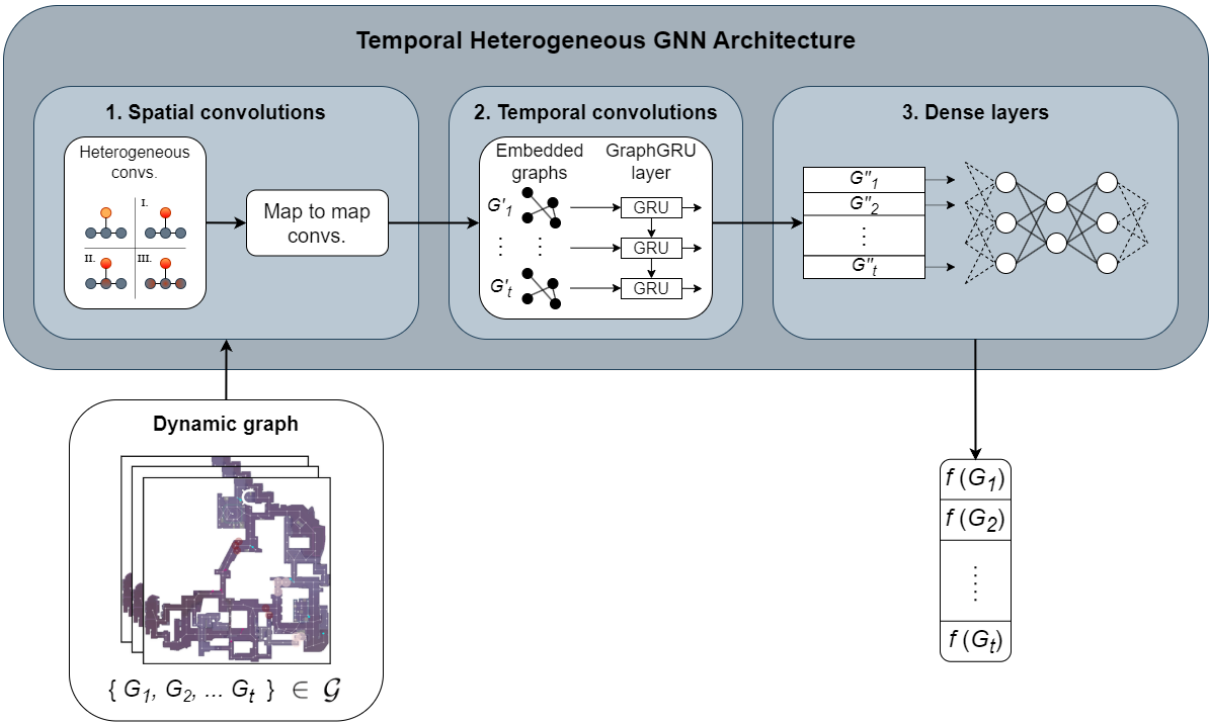


Figure 3.1: Architecture of the proposed THGNN. G_i represents a static graph from the input dynamic graph, G'_i is G_i after the spatial convolutions, G''_i is the flattened vector of G_i after the GRU layers and $f(G_i)$ is the predicted win probability for the defender side for G_i, where $i \in \{1,\ldots,t\}$ and t is the number of snapshots in the dynamic graph.

3.2. Spatial convolution strategy

The convolutional strategy used in the first part of the proposed network is based on the intuition that players are the primary sources of all changes in win probability, as they drive the events that influence it. This strategy is implemented through heterogeneous convolution layers, which are defined as an ordered sequence of convolution operations applied over specific edge types within the graph. The implemented layer following this strategy consists of three steps:

1. The initial step of the heterogeneous convolution layer involves the player nodes generating their own embeddings through the *player-is-player* self-loop edges. This step effectively performs dimensionality reduction, allowing the network to learn a more compact representation of the player features instead of manually selecting them.

2. In the second step, the information stored in the player embeddings are propagated to the nearest map nodes through *player-to-map* edges. This operation creates a map embedding that integrates both map-specific and player-related information.

3. Finally, the third step involves propagating the updated map embeddings among the map nodes via *map-to-map* edges.

The complete, ordered sequence of operations within a single heterogeneous convolutional layer is illustrated in Figure 3.2, providing a clear visual representation of the information flow across the graph.

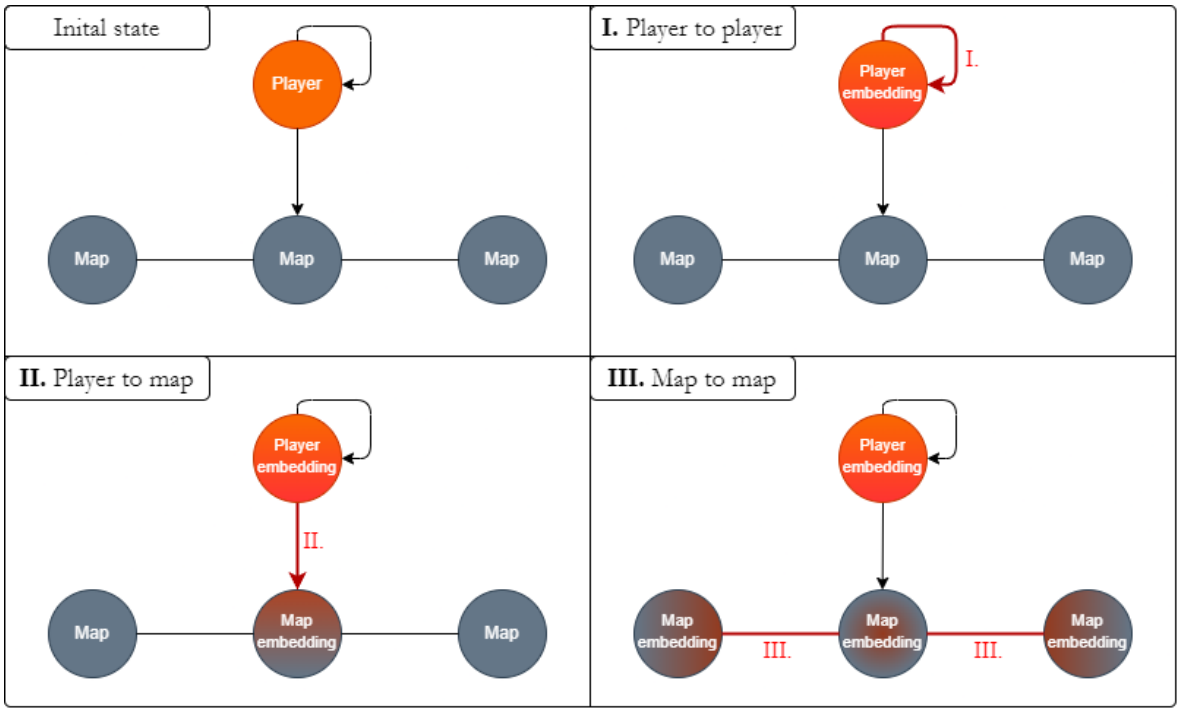


Figure 3.2: An example of a single layer of the described heterogeneous convolution applied on a graph. At each step, the edge over which the actual sub-convolution is applied is highlighted in red.

Following the heterogeneous convolution layers, additional convolutions were applied along the *map-to-map* edges to further propagate player-related information across the map. The number of layers for both convolution types was treated as a hyperparameter to be optimized. The best-performing model employs single-headed Graph Attention (GAT) [9] convolution for each sub-convolution. This architecture begins with two heterogeneous convolution layers, followed by three *map-to-map* convolution layers.

Model	Log-Loss	F1-score	AUC
Baseline	0.693	0.5	0.5
Logistic Regression	0.4688	0.7387	0.7564
XGBoost (proposed by P. Xenopoulos [1])	0.5353	-	0.7913
Proposed THGNN	**0.4587**	**0.7463**	**0.8549**

Table 3.1: Performance of the proposed THGNN compared to other models, evaluated across different metrics.

3.3. Results

After extensive hyperparameter optimization, the best-performing THGNN model was successfully trained, achieving an accuracy exceeding 76%. To validate the results of the THGNN, comparative models were trained using the tabular tracking dataset. These included a baseline model based on random predictions, a logistic regression and the XGBoost model proposed by Peter Xenopoulos [1]. Unlike the THGNN, which leverages temporal dependencies across snapshots, these tabular models generate predictions independently for each snapshot, without considering prior states. The training results, including the comparison of the THGNN with these baseline and tabular models, are summarized in Table 3.1.

The evaluation of the models was based on log-loss, F1 score, and AUC. The results indicate that the proposed network outperforms all other models across these metrics. In addition to these, it is essential to examine the model's calibration [6], as the model's primary purpose is probability estimation. Proper calibration is vital to ensure that the model contributes meaningfully to understanding the game dynamics. This requires reliable and interpretable probability outputs. The calibration curve of the THGNN (displayed in Figure 3.3) demonstrates that the model's probabilities are well-calibrated, which is further reinforced by its low Expected Calibration Error.

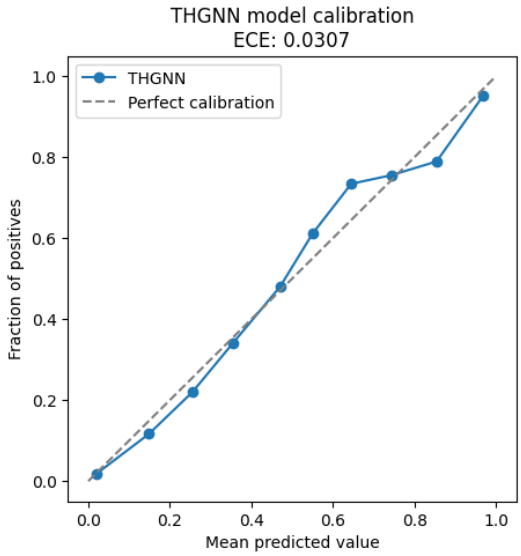


Figure 3.3: Calibration curve and Expected Calibration Error (ECE) of the proposed network.

4. Evaluating player actions

Explainability is a crucial aspect of artificial intelligence, especially in the field of sports analytics, where meaningful insights can drive player development and strategic improvements. Understanding the "why" behind model predictions is as important as the predictions themselves for the coaches and teams. Thus, in this section, we propose a technique to interpret the predictions of the THGNN model and leverage probability fluctuations to evaluate player actions. This approach aims to bridge the gap between complex AI models and practical, human-understandable insights.

4.1. Event-based explanation approach

To analyze the actions driving the probability fluctuations, a unique approach is proposed - one that focuses not on why the model predicts certain probabilities but on how fluctuations in these values correlate with player actions. This aligns with professional analysts' goals of identifying successful and poor decisions in gameplay.

Let $W(t)$ denote the estimated win probability for the defenders at timestamp t and $\Delta W(t-1,t)$ represent the change of the winning chance between timestamps t – 1 and t. This is formally calculated as shown on Equation 4.1.

$$\Delta W(t-1,t) \;=\; W(t) - W(t-1) \tag{4.1}$$

Player actions are the primary drivers in the win probability changes. Thus, $\Delta W(t-1,t)$ can be expressed as the cumulative impact of all actions performed by all players between timestamps t – 1 and t. This can be expressed as shown on Equation 4.2

$$\Delta W(t-1,t) = \sum_{p=1}^{10} \sum_{a=1}^{n_p} \delta W_{a,p}^{[t-1,t]} \tag{4.2}$$

where p denotes the player index, n_p is the number of actions done by player p between timestamps t – 1 and t, and $\delta W_{a,p}^{[t-1,t]}$ represents the probability change caused by the single action a by player p that happened between t – 1 and t. In the above equation, the $\delta W_{a,p}^{[t-1,t]}$ individual effects of actions are not determined directly - only their combined impact is known. However, by using a local explainability model and Shapley values [8], it is possible to identify the contribution of each action.

4.2. Event datasets

To compute the direction and magnitude of each action's impact on the win probability, an event dataset is created that captures the events of the selected round for analysis. This was achieved by extracting state-describing features of players and the map nodes from the static graphs that compose the dynamic graph. These features include player-specific attributes such as position, speed, view direction, and active weapon, as well as map-specific conditions like whether an area is burning or smoked. Only state-describing features were considered, as these are directly or indirectly influenced by player decisions. As a result, the event dataset was structured so that each feature f represented a specific state-describing feature of either a player or a map node. The t-th record in this dataset captured the changes of all f features between consecutive timestamps $t-1$ and t within the round. An additional column is then added to this table, representing the win probability change between timestamps $t-1$ and t, which served as the output variable for the explainability model.

The event dataset only captures actions occurring between consecutive snapshots. However, the THGNN model was trained on dynamic graphs representing 5-second intervals, meaning that actions taken seconds before a specific timestamp t could still influence the win probability at t. This event dataset does not yet account for temporal patterns spanning the entire 5-second interval. To address this limitation, multiple event datasets were created with varying time differences: one capturing consecutive events and others aggregating events over 1, 2, 3, 4, and 5-second intervals. By training

separate explainability models on each of these datasets and aggregating their results, it becomes possible to analyze and interpret the cumulative impacts of actions from different parts of the time interval, providing a more comprehensive understanding of how player actions influence win probability changes.

4.3. Explanation models and Shapley values

Explainability models were trained on the event datasets to capture the relationships between player actions and changes in win probability across varying time intervals. Ridge regression models with α values of 0.1 were employed, intentionally overfitting the datasets to replicate the GNN's predictions as closely as possible for the selected round. Subsequently, the SHAP Python package was utilized to estimate the Shapley values, quantifying each feature's contribution to the win probability change. This process provided an initial set of SHAP-based explanations.

However, post-processing was necessary to refine these explanations and align them with the domain requirements. A filtering mechanism was applied to the Shapley values of the features to ensure their relevance to the context, adhering to the following conditions:

1. A player eliminated during the round cannot impact the win probability after their death. To address this, a 2-second time window following a player's elimination was allowed, during which contributions could still be attributed[4]. After this period, the player's Shapley values are zero.

2. If a state-describing feature f does not change during the analyzed interval, its contribution to the win probability is zero.

The Shapley analyses for the various time intervals were passed through this filter to ensure domain-relevant explanations. Following this, the Shapley values for each feature were averaged across timestamps, aggregating contributions from different intervals. The resulting dataset provided, for any given time t, Shapley values for each feature based on the preceding 5 seconds, reflecting their impact on the win probability. Using this dataset, the impact $\delta W_{a,p}^{[t_1,t_2]}$ of any specific action a by player p between timestamps t_1 and t_2 could be calculated with the formula shown in Equation 4.3

$$\delta W_{a,p}^{[t_1,t_2]} = \frac{S_a}{\sum_f |S_f|} \Delta W(t_1, t_2) \tag{4.3}$$

where S_a is the Shapley value of the action a, $\sum_f |S_f|$ is the sum of the absolute values of all Shapley values for the features, and $\Delta W(t_1, t_2)$ is the win probability change between timestamps t_1 and t_2. As a result, each action carried out by the players is evaluated by the direction and magnitude it affected the team's winning chances.

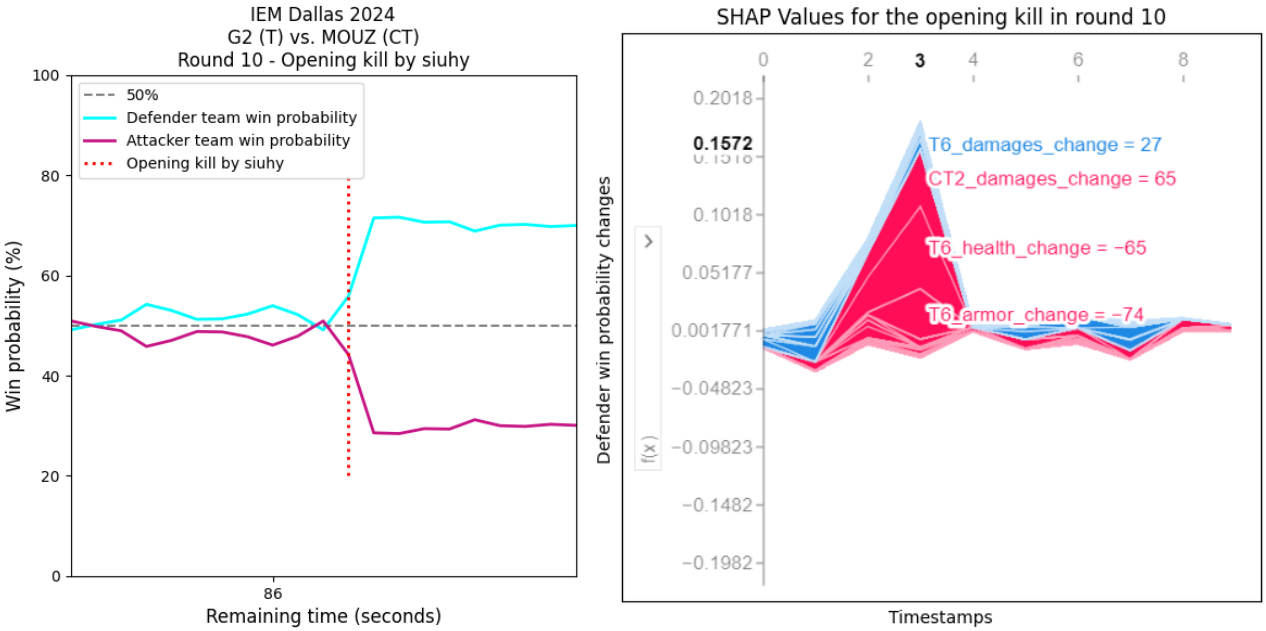
[4] In some cases, players with low health deliberately sacrifice themselves to give their teammates a strategic advantage.

5. Use cases

This section highlights various use cases where the proposed framework can provide deep analytical insights to professional teams in this esport. By leveraging the framework, teams can gain information about important actions to better analyze their players.

5.1. Highlighting key events

The proposed framework enables the analysis of a selected round by highlighting its key events - those critical moments that result in the most significant changes in win probability. These highlighted events represent the pivotal actions that shaped the outcome of the round. The explanation framework provides the tools to identify and interpret these impactful moments. Figure 5.1 visualizes an example illustrating an elimination event.

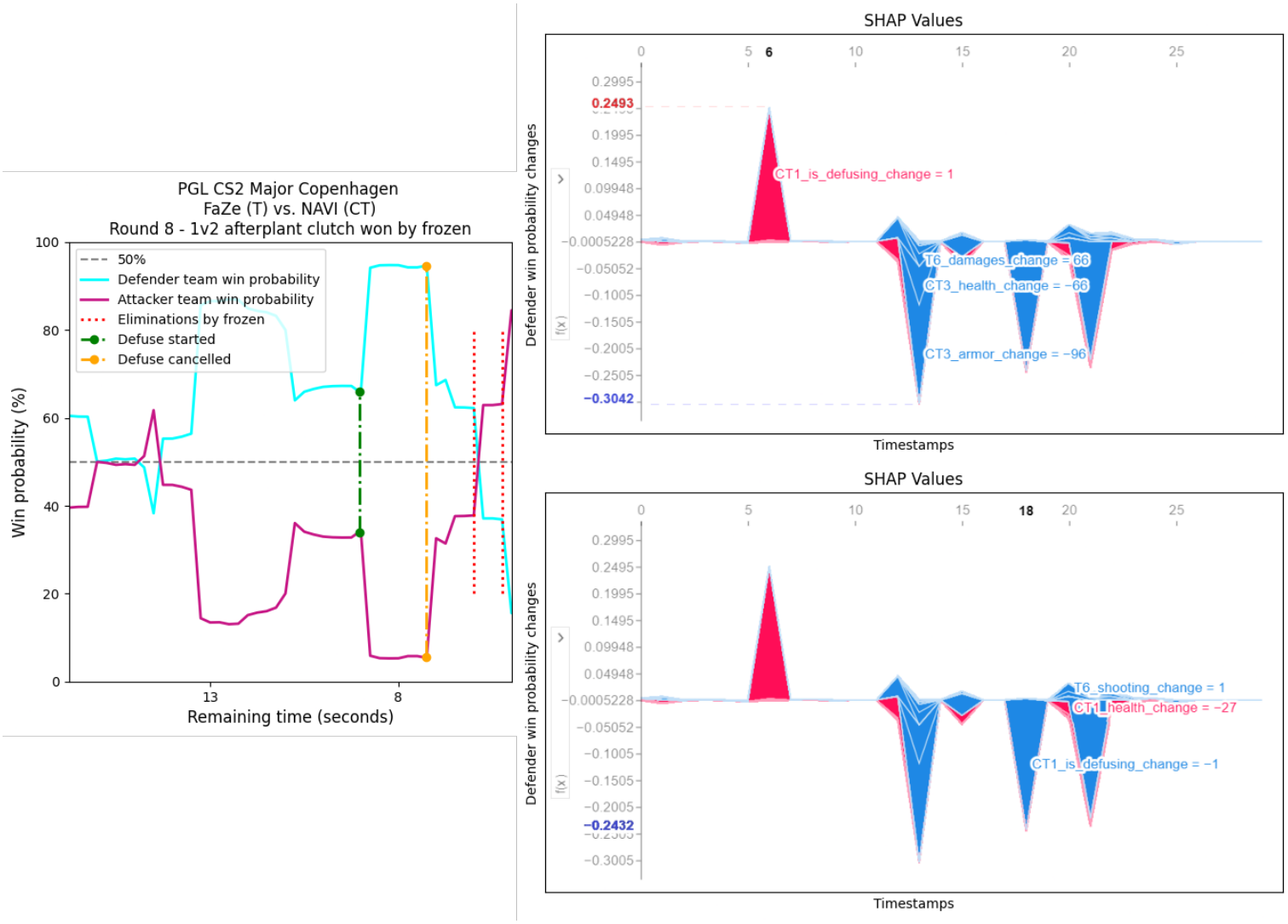


(a) (b)

Figure 5.1: An opening elimination event from IEM Dallas 2024 from a match between teams G2 and Heroic. (a) The win probability predicted by the proposed model. (b) Features with the highest impact corresponding to the opening kill and their effect on the win probability. The x axis represents the time stamps, while the y values show the probability changes between snapshots normalized between 0 and 1. Features highlighted in red have a positive effect on the defenders' win probability, while those in blue indicate events that drive the winning chance in a negative direction.

The analysis shows that the opening kill successfully secured by siuhy, a member of the defending team, increased the defender's win probability by 15%. The SHAP values demonstrate that the framework is capable of recognizing this. The features with the highest Shapley values are related to the attacker called Stewie2K (identified by T6) losing his life and siuhy's increased dealt damages statistic. Another analytical example can be seen in Figure 5.2. In this scenario, a player from the attacking FaZe team called frozen, won a 1v2 clutch situation from a disadvantageous position, successfully defending the planted bomb and winning the round. Alongside the kills, the model

correctly identifies the importance of other events depending on the context. With very little time left in the round and the defenders needing to disarm the bomb, the start of the 5 second long defuse significantly boosted their chances of winning (+25%). However, once the defuse was interrupted, this advantage quickly diminished, as reflected by the model's evaluation.



(a) (b)

Figure 5.2: A 1v2 clutch situation won by a player called frozen from the attacker team (FaZe) in the grand finals of the PGL Copenhagen major. (a) The probability predictions by the THGNN model, with the important events highlighted. (b) The result of the explanation analysis, highlighting the most important features with the highest impact on the win probability change.

The analyses presented so far, when compared with the broadcast footage of the matches, reveal that the explanations align with intuition, confirming the model's ability to highlight and explain significant events. These examples strengthen the trustworthiness of the model. One of the

advantages of the model, however, is that it is also capable of emphasizing seemingly unimportant actions that may appear to have lesser significance. An example of this is presented in Figure 5.3.

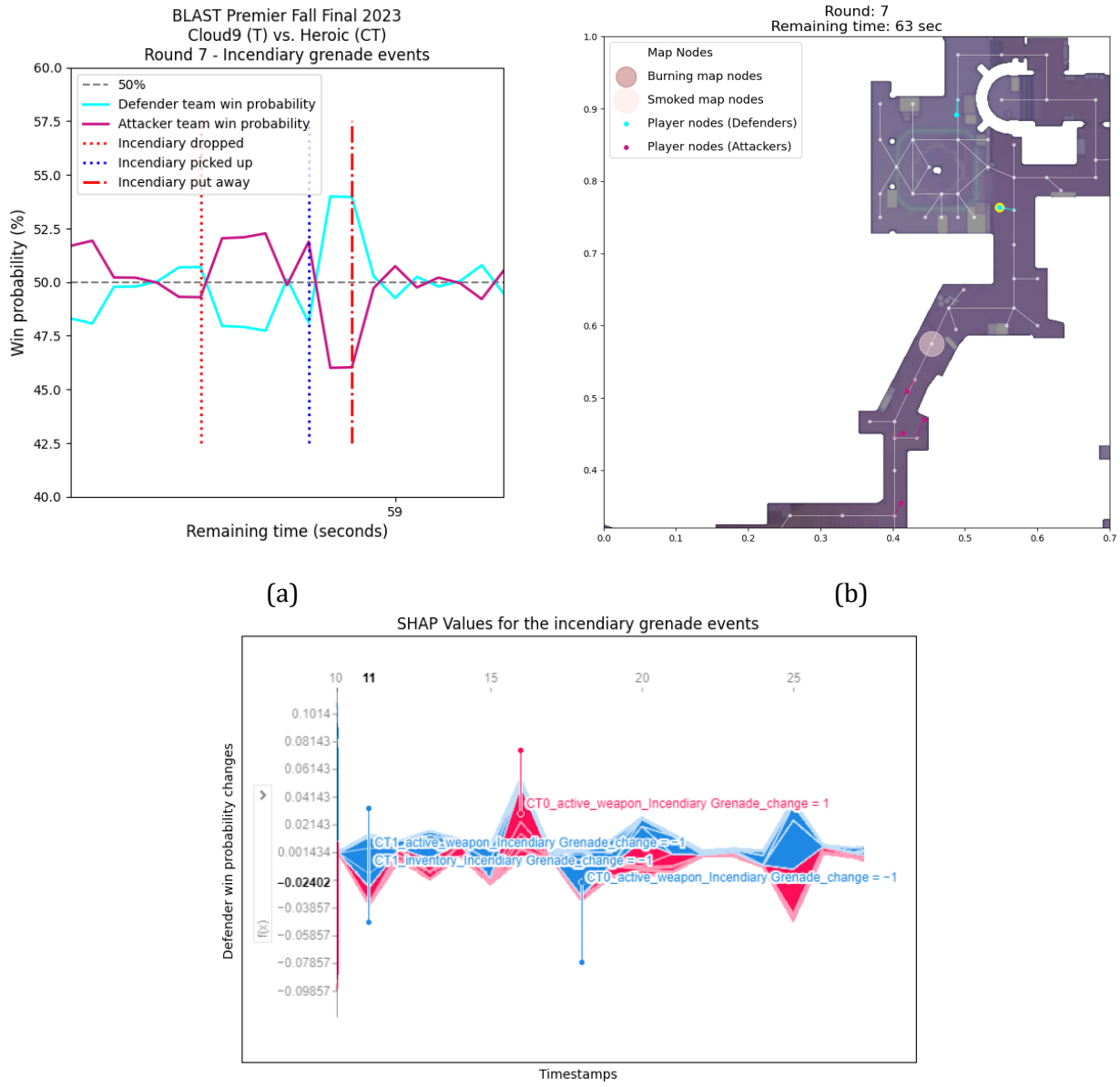
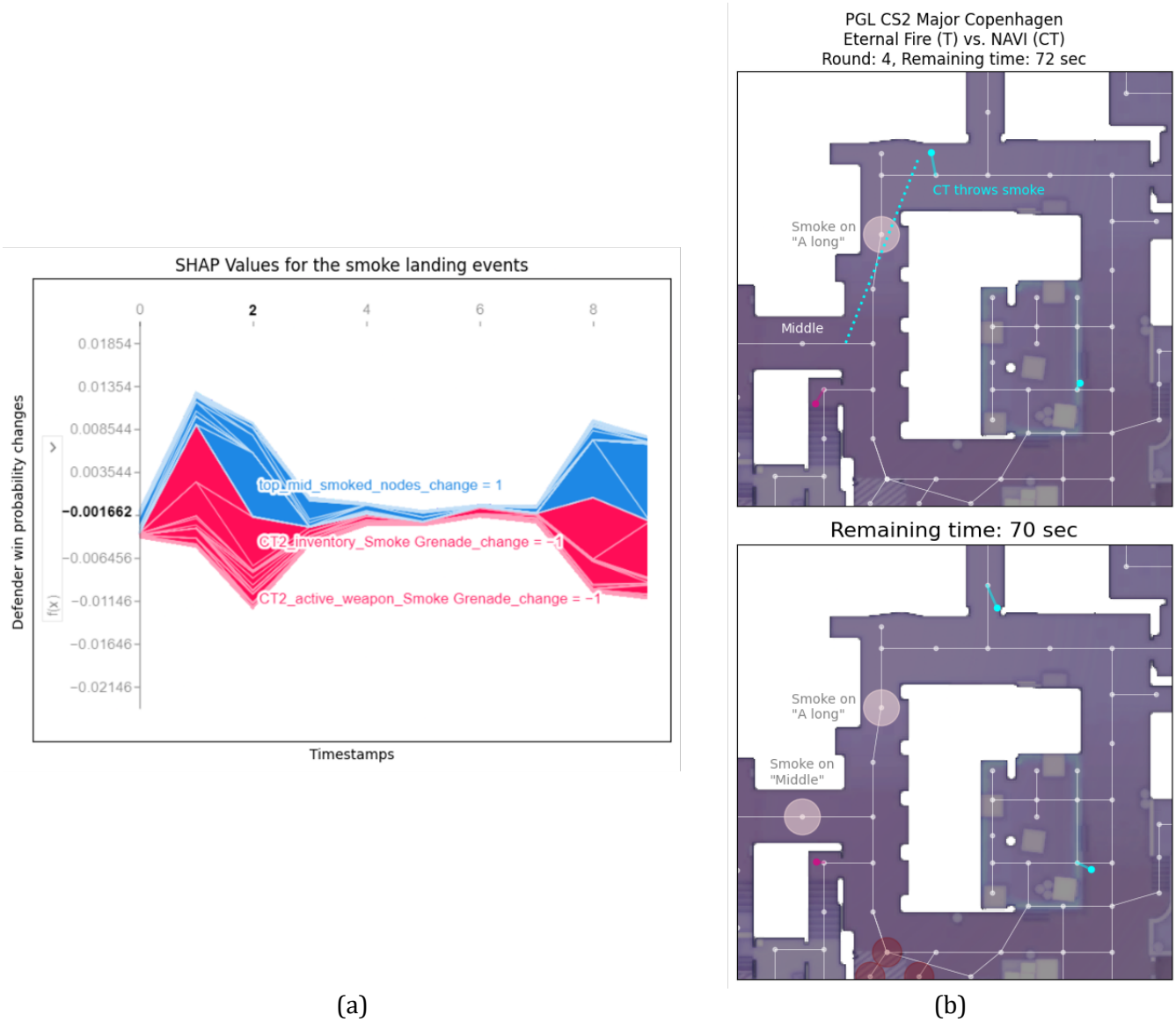


Figure 5.3: Impact of an incendiary grenade exchange on the CT win probabilities. (a) Probability predictions for the examined interval. (b) The game-state visualized for the examined situation. (c) Shapley values highlighting the importance of the incendiary grenade events for the win probability.

In this example, around the one-minute mark, the win probability for the Counter-Terrorists first decreases and then shows a noticeable increase for a brief period. While the cause of these changes is not immediately apparent from the match footage, the explanatory model identifies the key events behind them. At the 61-second mark, a defender drops an incendiary grenade, leading to a temporary

decrease in the win probability as the other defender, highlighted in yellow in Figure 5.3(b), is required to step back and retrieve it. Once the grenade is in his inventory and selected as the active weapon, the team's win probability increases by nearly 5%. Notably, four attackers are positioned nearby, preparing for a potential execution towards the bombsite the defenders guard. Had they decided to push, a well-timed incendiary grenade could have delayed their advance. Interestingly, the push does not occur, yet the model positively valued the grenade being held as an active weapon, recognizing the possible threat and the equipment's potential impact.



(a) (b)

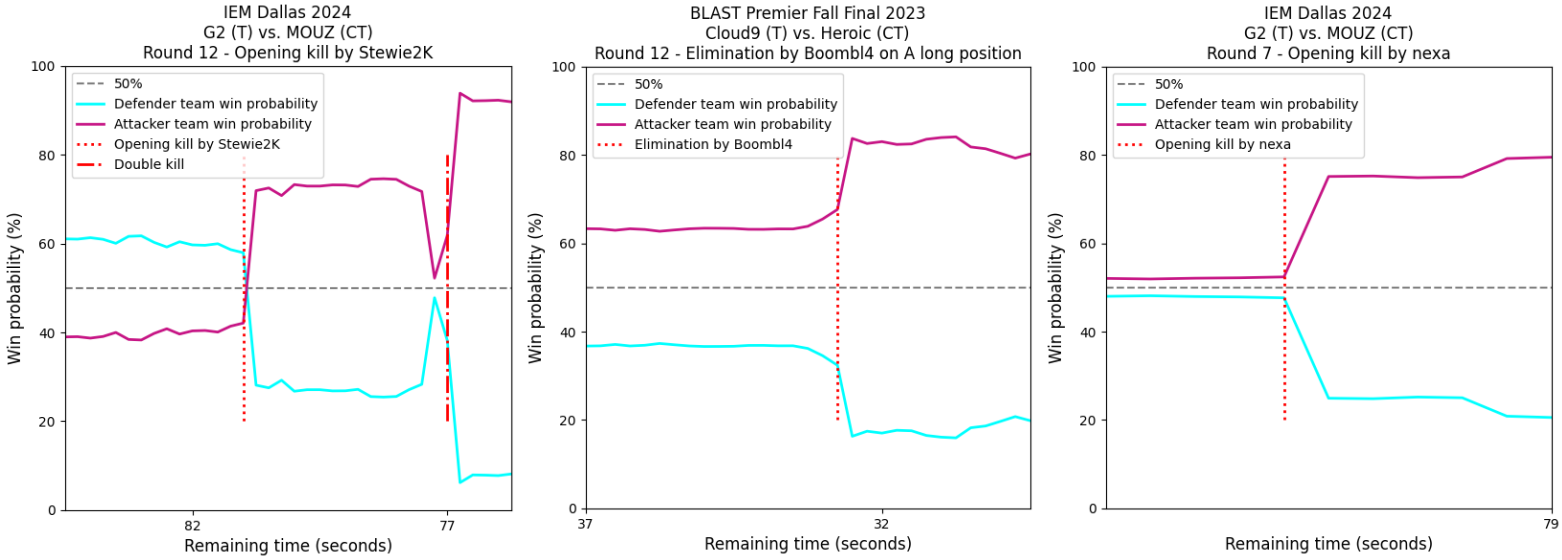
Figure 5.4: A defender balancing the win probability of his team by throwing a smoke grenade. (a) The SHAP values highlighting the most important events. (b) The game-states visualized at the moment of the grenade throw (top) and two seconds after (bottom).

Another analysis is presented in Figure 5.4, which further demonstrates the framework's ability to perform detailed evaluations. In this example, a smoke grenade thrown by the attacking team lands at a position known as A-long in the examined moment (visualized in Figure 5.4(b)). This is a common

tactic used by attackers to block the vision of potential defenders in that area and gain map control. A strategic countermeasure by the defender is to throw a smoke at the section of the map called middle, preventing the attackers from easily advancing and securing map control. The framework effectively recognizes the threat posed by the smoke landing on A-long, as shown by the SHAP values. However, the win probability does not change, as the defender immediately counters by throwing a smoke to the map area called middle, thus balancing the probabilities. The framework also highlights the defender's smoke throwing as an action with positive impact, which is reflected in the SHAP values. This highlights the framework's precision in assessing and comparing the significance of various in-game events.

5.2. Event impact comparison

One of the key advantages of the model is its ability to estimate round winning probabilities for the teams based on the specific context. As a result, the same events can have varying impacts (win probability changes) across varying situations. This allows for a deeper analysis of how similar actions influence outcomes in different contexts, prompting intriguing questions for professional players: why does an event have a greater or lesser effect across rounds, and what contextual factors contribute to these variations? A notable example is the opening kill: in professional Counter-Strike, securing the opening kill is crucial, as the team that first gains a man advantage has significantly more control over the round's outcome. In the example shown in Figure 5.1, the opening kill by siuhy increased the defending team's win probability by 15%. For comparison, Figure 5.5 illustrates several other opening kills from different matches and scenarios.



(a) (b) (c)

Figure 5.5: Opening eliminations across different games and scenarios. (a) An opening elimination by Stewie2K increased the attacker's winning chance by almost 30%. (b) Boombl4 from the attacking team creates an opening and gains an additional 17% winning chance for his squad. (c) An opening by the player called nexa increases attacker win chances by 22%.

In the first case, Stewie2K's opening kill resulted in a nearly +30% increase in his team's win probability. This significant increase can likely be attributed to the fact that his team was considerably underpowered in terms of weaponry compared to the opponents, making the opening kill far more impactful. Notably, a double elimination occurs shortly thereafter, further boosting the win probability by +30%. The fact that two eliminations provide a similar win probability increase as the crucial opening kill underscores the importance of these key moments in shaping the outcome of the round. An opening elimination with balanced team equipment values is illustrated in Figure 5.5(b). This event led to a +17% increase in the attacking team's win probability, significantly less as in the first scenario. Figure 5.5(c) analyzes another opening kill event in a round where both teams had balanced weaponry. However, in this case, the increase was 5% higher (+22%) compared to the previous. This can likely be attributed to the attackers being in a much stronger position to secure one of the bomb sites compared to the other attacking team in the (b) scenario, further amplifying the importance of this event.

These examples demonstrate that identical events can have varying impact values depending on the context, offering professional teams a valuable opportunity to examine and understand these dynamics. Additionally, this framework allows teams to experiment with strategic executions by testing minimal variations, receiving quantitative feedback on which slightly altered scenarios resulted in the greatest impact, such as for opening kills. This type of analysis can provide crucial insights and help teams fine-tune their strategies, offering a significant advantage during preparation.

6. Related work

The primary goal of this project is to develop a machine learning-driven player action evaluation system, building on the methodology introduced by Peter Xenopoulos [1]. Their approach utilized an XGBoost model trained on tabular data representing game state with plenty of features, such as player positions, weapons, health, time remaining, and number of players alive, to predict the winning probabilities for Counter-Strike rounds. By generating predictions at multiple time points, they analyzed fluctuations in team win probabilities over time, attributing these changes to player actions and in-game events. They also proposed a novel player rating metric, Win Probability Added (WPA), which quantified a player's impact on their team's win probability by summing the effects of damage-related events within a round. This innovative approach provided a focused evaluation of players' contributions through their damage impacts, showcasing the potential of win probability-based metrics in assessing performance. Our solution improves upon the previous approach by using THGNNs to capture both the spatial structure of the map and temporal patterns in player actions, allowing for a more accurate and nuanced evaluation of player performance. In addition to these improvements, our framework is also capable of evaluating the significance of non-damage related events, providing a comprehensive assessment of player actions.

7. Conclusion

The presented work introduces a novel approach for evaluating player actions in professional Counter-Strike 2 using Temporal Heterogeneous Graph Neural Networks. This framework predicts round-winning probabilities at a frame-by-frame level, providing a detailed analysis of player actions and their impact on match outcomes. By incorporating both spatial and temporal graph convolutions, the model captures complex interactions between players and their environment, offering insights into the dynamics of the game.

We propose an explanation technique to examine fluctuations in win probability through Shapley values. These values attribute changes in win probability to specific player actions, such as eliminations, grenade usage, or positioning, giving coaches and analysts a deeper understanding of the effectiveness of each decision. This leads to more informed strategic planning, helping teams refine their tactics and focus on areas that most impact their success. Additionally, the framework can highlight critical moments in a round, such as opening kills or clutch plays, by measuring their effect on win probability. Teams can use these insights to better understand which actions are most influential in various situations.

In conclusion, this work offers a powerful tool (available at [10]) for analyzing and improving player performance in Counter-Strike 2. By explaining and quantifying the impact of individual actions, the framework enables teams to optimize strategies and gain a competitive edge in esports.

References

[1] Peter Xenopoulos, Harish Doraiswamy, and Claudio Silva. Valuing player actions in counter-strike: Global offensive. In Proceedings of the IEEE International Conference on Big Data, 2020.

[2] Peter Xenopoulos, Bruno Coelho, and Claudio Silva. Optimal team economic decisions in counter-strike, 2021. URL https://arxiv.org/abs/2109.12990.

[3] Hongyu Xu and Sarat Moka. Rating players of counter-strike: Global offensive based on plus/minus value, 2024. URL https://arxiv.org/abs/2409.05052.

[4] Peter Xenopoulos, Joao Rulff, and Claudio Silva. ggviz: Accelerating large-scale esports game analysis, 2022. URL https://arxiv.org/abs/2107.06495.

[5] NER0cs, "Introducing rating 2.1," *HLTV.org*, 14-Oct-2024. [Online]. Available: https://www.hltv.org/news/40051/introducing-rating-21. [Accessed: 29-Nov-2024].

[6] Chuan Guo, Geoff Pleiss, Yu Sun, and Kilian Q. Weinberger. On calibration of modern neural networks. In Proceedings of the 34th International Conference on Machine Learning (ICML), 2017.

[7] Youngjoo Seo, Michaël Defferrard, Pierre Vandergheynst, and Xavier Bresson. Structured sequence modeling with graph convolutional recurrent networks, 2016. URL https://arxiv.org/abs/1612.07659.

[8] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017. URL https://arxiv.org/abs/1705.07874.

[9] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks?, 2022. URL https://arxiv.org/abs/2105.14491.

[10] Counter Strike 2 player action evaluation framework, 2024. GitHub. https://bit.ly/3Zu78tz [Accessed: 29-Nov-2024]

Appendix

A.1 Full round predictions

In the Use Cases section, several situations were analyzed in detail, with a focus on specific events. This part of the appendix illustrates some examples of the win probabilities visualized for full rounds.

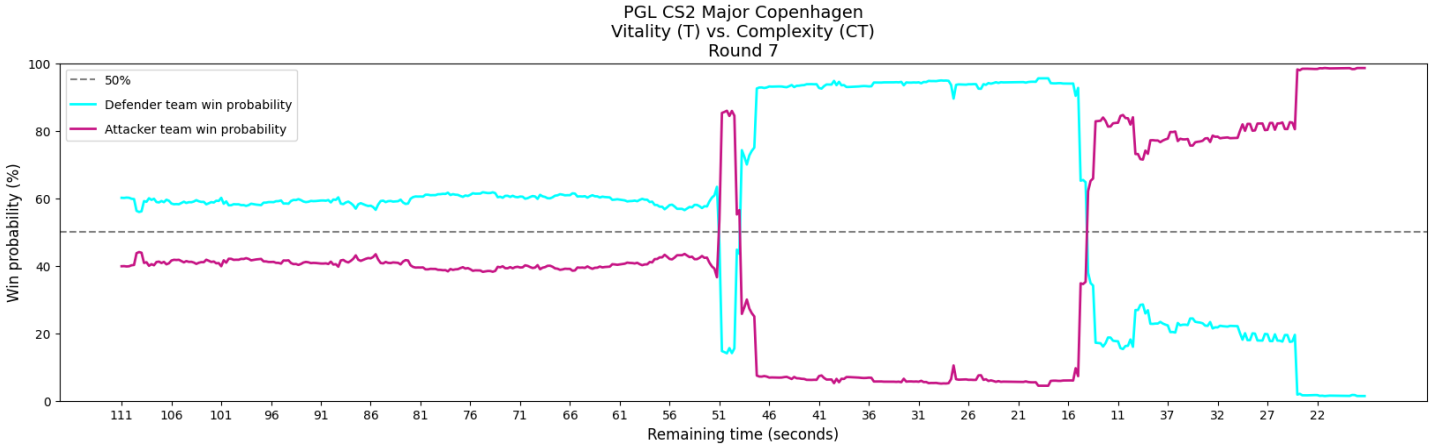


Figure A.1.1: Estimated win probabilities for the teams in round seven at a PGL CS2 Major Copenhagen match between teams Vitality and Complexity.

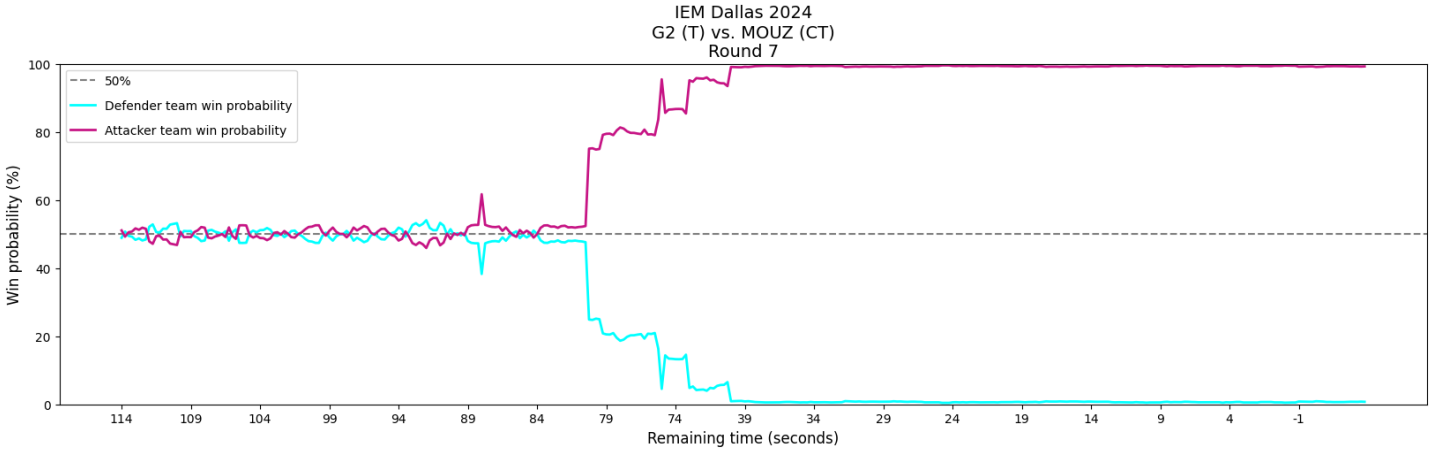


Figure A.1.2: Estimated win probabilities for the teams in round seven at the IEM Dallas 2024 event in a match between teams G2 Esports and MOUZ.