Abstract—The multiplayer online battle arena (MOBA) games have become increasingly popular in recent years. Consequently, many efforts have been devoted to providing pregame or in-game predictions for them. These predictions can be used in many MOBA esports-related applications, such as artificial intelligence commentator systems, in-game data analysis, and game-assistant bots. However, these works are limited in the following two aspects: the lack of sufficient in-game features and the absence of interpretability in the prediction results. These two limitations greatly restrict the practical performance and industrial application of the current works. In this work, we collect a large-scale dataset containing rich in-game features for the popular MOBA game Honor of Kings. We then propose to predict four types of prediction tasks in an interpretable way by attributing the predictions to the input features using two gradient-based attribution methods: Integrated Gradients and SmoothGrad. To evaluate the explanatory power of different models and attribution methods, a fidelity-based evaluation metric is further proposed. Finally, we evaluate the accuracy and fidelity of several competitive methods to assess how well machines predict events in MOBA games.

Index Terms—Evaluation method, game prediction, interpretable, multiplayer online battle arena electronic sports (MOBA eSports).

I. INTRODUCTION

NOWADAYS, with the fast development of the gaming industry, electronic games are becoming increasingly popular, generating huge amounts of profit. Among all the genres of electronic games, MOBA games are one of the most popular and highest-grossing types, such as Defense of the Ancient II (DotA2), League of Legends (LoL), and Honor of Kings (HoK). Together, these three popular MOBA games have more than 300 million monthly active players globally and an even larger potential audience among the streaming media community. Along with MOBA games’ flourishing, much research has been done to predict the results before and during the games. These studies can be categorized into two types: pregame predictions that predict the results of MOBA games before they begin [1], and in-game predictions that predict according to the in-game situations of the games [2]. These predictions can provide more information to the audience and the commentators, and therefore, can be used in many MOBA-related applications, such as the artificial intelligence (AI) commentator systems, the in-game data analysis, and the game-assistant bots that can provide basic analysis and suggestions for the players.

In many senses, the in-game predictions are more significant and have broader application scenarios than the pregame predictions. However, although some progress has been made, existing studies for in-game predictions are limited in the following two aspects. The first limitation is: the insufficiency of large-scale in-game data. Due to the difficulty in collecting in-game data, current datasets for MOBA game predictions contain only pregame features [3] or limited types of in-game features such as “gold,” “experience,” and “death” [2].

Second, the predictions of the current works are noninterpretable, which greatly limits their application. The term “noninterpretable” means that the predictions are given without showing the underlying reasoning why these results are achieved by the prediction models. By showing what game features in the current input are contributing to the prediction result, the researchers and the audience can get more information that is helpful to understand the games. As one can see in Fig. 1, the noninterpretable model used in LoL 2019 World Championship can only give the winning probabilities of the two teams (80% versus 20%), which is opaque to the audience. The audience
can understand the game situations better if they are shown what factors result in these predictions. For example, if Team-1 gets a larger winning rate and the feature “campMoneyDiff” (the difference between the two team’s gold-amount) is one of the key features to the win prediction, then Team-1’s superiority is (partly) due to its lead in gold amount. Moreover, if a significant change appeared in the predictions within a short period (such as 5 s), then the top-contributing features will explain why this sudden breakthrough happens. From the perspective of both the researchers and the audience, this additional explanatory information is valuable and helpful to understand what is actually going on in the match, as human’s analyses may not always be consistent with the prediction models’ attributions, and when this happens, the interpretable results can give more information to the researchers and the audience. Hence, there is a need to generate human-interpretable predictions for MOBA games.

In this work, to facilitate the study of in-game predictions for MOBA games, we collect a large-scale dataset that contains in-game records with rich features extracted from the game-core data (the back-end data of HoK) of 50 278 games. Every second, we record more than 2000 features. Within this dataset, one can easily train models to predict the important events of HoK.

Given the game records as input, we further train two state-of-the-art (SOTA) sequence modeling networks, long short-term memory (LSTM) network [4] and Transformer [5], to predict the important events in HoK. However, as noted previously, these well-known black-box networks are non-interpretable, so one cannot understand the reasons for their predictions. To mitigate this problem, we propose to apply two gradient-based attribution methods, Integrated Gradients (IG) [6] and SmoothGrad (SG) [7], to interpret the prediction result by attributing it to the top-contributed feature dimensions. As reasoned by the authors in [6] and [7], with these methods we can determine which features primarily contribute to the current prediction result.

Although the attribution methods can interpret the predictions of deep neural networks, there is another issue in that it is difficult to evaluate the correctness of the attribution methods. In other words, we can hardly know how well they interpret the prediction results. So, inspired by the fidelity-based evaluation methods in Natural Language Processing [8], [9], we propose the fidelity metric to evaluate the explanatory power of attribution methods and prediction models. The main idea of fidelity is: when extracting the top-contributed feature dimensions using an attribution method, if those extracted features have the potential to construct an optimal proxy model that agrees well with the original model on making a prediction, then this attribution method is good. In other words, we can evaluate the attribution results by measuring the consistency between the proxy model’s prediction results and the original model’s predictions: the more consistent these results are, the better the explanatory power of the attribution method will be. In experiments, the fidelity metric proves that the IG method interprets better than the SG method in MOBA game event prediction tasks.

In summary, our contributions are threefold.

1) We propose to predict four representative tasks (“win,” “Tyrant,” “kill,” and “be-kill”) based on a large-scale HoK dataset.

2) We achieve interpretable event predictions with two SOTA sequence modeling networks and two SOTA gradient-based attribution methods. These works can serve as strong baselines for this task in future studies.

3) We evaluate the explanatory power of attribution methods and prediction models, proposing the fidelity metric to quantitatively measure how well they interpret the prediction results.

II. RELATED WORKS

A. MOBA Game Prediction

Studies on predicting MOBA games consist of pregame and in-game predictions. They can also be divided into the outcome (“win”) predictions and the event predictions. Pregame (outcome/“win”) predictions focus on training prediction models based on pregame features such as hero-selections and players’ historical records. Among these works, [1] is the first to predict the DotA2 results before the games start. Kalyanaraman [10] follows this work by combining the genetic algorithm with logistic regression (LR) and reports a higher prediction accuracy. Kinkade et al. [11] propose to build prediction models for DotA2 based on two different sets of training data: one comprising only the hero-selection information, and the other consisting of the full postgame data. The authors in [3], [12], and [13] further evaluate the performances of several machine learning methods for DotA2 win predictions, including LR, Naive Bayes, gradient boosted decision trees (GBDT), and other methods. Song et al. [12] also use “hero-lineups” and LR to predict the winning side of DotA2. Semenov et al. [3] evaluate the performance of several machine learning methods such as Naive Bayes and LR based on the hero selections in Dota2. Makarov et al. [13] predict the winning teams and their winning probabilities for Dota2 and another first person shooter (FPS) electronic game, Counter-Strike: Global Offensive (CSGO). Wang al.[14] propose to improve pregame DotA2 prediction using a better representation of hero-selection information.

Although much research has been done to perform pregame predictions for MOBA games, in-game predictions are more informative and useful. Therefore, recent research focuses more on in-game MOBA game outcome and event predictions. Yang et al. [2] first introduces three in-game features to achieve win predictions for Dota2. Hodge et al. [15] use machine learning methods such as LR and decision tree to predict the results of Dota2 using professional-level in-game data. Yang et al. [16] propose a two-stage model TSSTN to perform interpretable in-game predictions for HoK. However, this model can attribute the results to only six human-selected features such as “gold” and “heroes.” Moreover, their work achieves interpretability at the cost of accuracy, which undermines the performance. The SHAP method proposed in [17] can also give interpretation for the prediction procedures of the models. Value-based reinforcement learning methods [18]–[21] can also give in-game win predictions without interpretability. The authors in [22] and [23] try to predict the results of LoL using multiple data collected from sensors and other hardware. For event predictions, [24]...
aims to predict the events inside the games such as hero deaths. Instead of predicting the outcome of esports matches, this work focuses on the “micropredictions” in the games, in particular, the predictions for the heroes’ death.

B. Gradient-Based Attribution Methods

In order to attribute the prediction of a deep network to its input features, Sundararajan et al. [6] propose a gradient-based attribution method IG. Two fundamental axioms, Sensitivity and Implementation Invariance, are also proposed to prove the correctness of IG. Following this work, He et al. [25] apply IG to measure the word-importance for neural machine translation. Chen et al. [26] also apply IG to interpret the results of image-recognition networks.

Smilkov et al. [7] propose another gradient-based attribution method, SG, to identify pixels that strongly influence the final decisions of image classifiers. By adding noise to the original image, we get a set of similar images. Then, by averaging the gradients for each image to the image classifiers’ outputs, we can get a better sensitivity map (attribution result) of the original image to the classification result.

C. Fidelity Metric for Explanatory Power

The idea of “fidelity” is primarily used in the domain of model compression [27], [28] and model distillation [29], [30]. Recently, Jacovi and Goldberg [9] propose to utilize a similar concept of “faithfulness” to evaluate the interpretable methods for deep-learning-based nature language processing (NLP) models. Li et al. [8] further propose a fidelity-based metric and its practical approximation method for neural machine translation.

D. Explainable Artificial Intelligence (XAI)

With the purpose of explaining the autonomous decisions and actions of the artificial intelligence models to human users, much research effort has been devoted to finding the rationale for models’ decisions. Among these researches, [31]–[39] give thorough surveys on this topic. While many XAI methods can be applied to the esports-prediction scenarios, we choose to study the gradient-based methods as our attribution methods and their evaluation due to the progress schedule limit of our experiments. We believe further research can be done to promote this area.

III. TASK AND DATASET

A. Prediction Tasks

We propose to predict four important events for HoK games: “win,” “Tyrant,” “kill,” and “be-kill.” These are four of the most important events in HoK and other MOBA games. The descriptions of these four events are as follows.

1) Win: Predicting which team will win the game. The “win” prediction task is the most generally acknowledged and the most studied among all the tasks. Throughout all the matches, the audience and the players care most about which team will win and why.

2) Tyrant: Predicting which team will seize the Tyrant. The “Tyrant” task is representative of all the tasks to predict which team will kill a boss monster.

3) Kill: Predicting who will be the next killer (the hero who kills the enemy). The event that one hero kills an enemy is one of the major focuses in MOBA games.

4) Be-kill: Predicting which hero will be killed next. The death of a hero may greatly affect the situations and results of the games.

B. Events Extraction

To predict the four events mentioned previously, we record the death information of all the heroes, monsters, and towers of 50 278 HoK games. The data are collected from the top 1% ranked (the “Conquerors” level) players’ games happened in the first quarter of 2020. The death information contains the death-frame (game-time), killer-information, hurt-information, and some other useful information, as shown in Fig. 2. Then, the task labels can be extracted from this death information. It is worth noting that one may extract more potential events from the death information, such as towers’ destruction and the next optimal equipment.

C. Feature Extraction

In addition to the death information, we also record more than 2000 in-game features every second of the games. We classify these features into five categories: “hero,” “global,” “monster,” “soldier,” and “tower.”

1) Hero: Heroes are the game characters that are controlled by the players. As shown in Fig. 3, hero features contain the information of the ten heroes in the game, including the hero’s ID (name), camp, level, kill count, assist count, death count, skills’ information, and many other features.

2) Global: Global features describe the game’s overall situations, including the game time, number of two camps’ alive heroes, money amount, and number of alive towers. An example of global features is shown in Fig. 4.

3) Monster: Monsters refer to the neutral creatures in the Wild. By killing the monsters, heroes can obtain gold,
experience, and buff (for some special monsters). Monster features contain the information of up to 27 monsters (Tyrant is one of them), including the monster’s health points (hp), alive-or-not status, location, attack, and monster type, as shown in Fig. 5.

4) **Soldiers**: Soldiers are the game units of the two teams that automatically generate and move to attack the rival heroes, tower, and base. By killing the rival soldiers, the heroes will get gold and experience. As shown in Fig. 6, soldier features cover the information of up to 82 soldiers. Features in this category include the soldiers’ camp, location, hp, alive-or-not status, soldier type, and attack.

5) **Towers**: Towers refer to the defense towers of the two teams that stand in the three lanes and around the base. They would automatically attack the rival soldiers and heroes that get near them. Tower features contain the information of two camps’ 22 towers, including the tower’s attack range, location, camp, distance to heroes, hp, tower type, and attack. An illustration is shown in Fig. 7.

### IV. Prediction

In this section, we first encode all the categorical features in the dataset into one-hot vectors, then concatenate them with other numerical features as the input vectors. Given the encoded input feature, we train two SOTA sequence modeling networks, LSTM and Transformer, to predict the occurrences of the aforementioned events.

#### A. Input Feature

Some of the in-game features in the collected dataset are categorical, such as hero-ID, skill-ID, and NPC-type. To better represent these categorical features, we encode them to one-hot vectors. For “Tyrant,” “kill,” and “be-kill” tasks, we choose the training data at $S$ seconds intervals before the event’s happening time (using the data from game-time $t - l + 1$ to $t$ to predict the event at $t + S$); for “win” task, we set some fixed time intervals, and choose the game records at these intervals as the training data.

#### B. Prediction Model

To capture the time-sequential characteristics of the games, we use SOTA sequence modeling networks such as LSTM [4] and Transformer [5] to perform the prediction tasks. A fully connected (FC) layer is used to make the final predictions for the
four tasks, as shown in Fig. 8. We use the same model structures for the tasks with the same input and output format, such as the “kill” and the “be-kill” tasks. For explicitness, we train a separate prediction model for each experimental setting.

V. ATTRIBUTION METHOD

In order to find the underlying reasons for the prediction models’ results, we utilize two gradient-based attribution methods, IG [6] and SG [7], to interpret the event predictions by attributing the prediction results to the input features. Specifically, IG fulfills this task by calculating the straight-line path integral of the gradient from a baseline input $X'$ to the current input $X$, while SG performs attribution by averaging the gradients of a set of similar inputs generated by adding Gaussian noise to the original input. However, there is a challenge in that the categorical features cannot be logically divided or added with noise. Therefore, to apply these attribution methods in our task, we deliberately design an additional embedding layer that maps the categorical features into continuous representations.

A. Integrated Gradients (IG)

The IG method was first proposed by [6] to attribute the results of deep networks to the input features. In this work, we use IG to find the top-contributing feature dimensions for the four prediction tasks in MOBA games. Specifically, let $x^t = [x^t_1, ... , x^t_n]^T$ be the input vector at game time $t$ and let $X = [x^{t-1+1}, ..., x^t]^T$ be the l-second sequential input up to game time $t$. Assume that $F$ is a prediction model, and $P(y|X)$ is its output. We set $X'$, which has the same dimension as $X$, to be the baseline, with all its elements to be 0. Then, the IG of $X$ is defined by the integral of gradient from $X'$ to $X$ in the straight-line path

$$IG_{i,j} = \int_{\alpha=0}^{1} \frac{\partial P(y|\tilde{X})}{\partial X_{i,j}} \bigg|_{\tilde{X}=X'+\alpha(X-X')} \, d\alpha$$  

(1)

where $IG_{i,j}$ represents the contribution of feature dimension $j$ to the prediction in $x^t$.

However, this theoretical formulation of IG is inconvenient for practical applications due to the existence of the path integral. A practical approximation of (1) can then be formulated by

$$IG_{i,j} \approx \frac{X_{i,j} - X'_{i,j}}{\text{steps}} \sum_{k=1}^{\text{steps}} \frac{\partial P(y|\tilde{X})}{\partial X_{i,j}} \bigg|_{\tilde{X}=X'+\frac{\alpha}{\text{steps}}(X-X')}$$  

(2)

where $\text{steps}$ is the number of steps that evenly distribute from the baseline $X'$ to the input $X$. The larger $\text{steps}$ we choose, the better approximation of IG we will get. In practice, $\text{steps}$ ranging from 100 to 300 results in good enough approximations and reasonable efficiency [6].

B. SmoothGrad (SG)

SG is also a gradient-based attribution method, first proposed by Smilkov et al. [7], which can be utilized to attribute the prediction to the input features for MOBA games. Assume that the output of the prediction network is $P(y|X)$, where $y$ is the target and $X$ is the input. Then, the SG for the $j$th dimension of $x^t$ in $X = [x^{t-1+1}, ..., x^t]^T$ is calculated by

$$SG_{i,j} = \frac{1}{\text{steps}} \sum_{k=1}^{\text{steps}} \frac{\partial P(y|\tilde{X})}{\partial X_{i,j}} \bigg|_{\tilde{X}=X+\mathcal{N}(0,\sigma^2)}$$  

(3)

where $\text{steps}$ is the number of generated samples and $\mathcal{N}(0,\sigma^2)$ represents the zero-mean Gaussian noise with standard deviation $\sigma$. $SG_{i,j}$ represents the contribution of feature dimension $j$ to the model’s output $P(y|X)$ in $x^t$.

C. Embedding

To make the categorical features in the input continuous, we need to transform the original input into embedding vectors first. As Fig. 9 shows, the input vector of 5885 dimensions will first be mapped into embedding vectors of 2001 dimensions, in which numerical features will be directly copied after normalization and categorical features such as hero and skill-ID will be transformed using several parallel FC layers. Input dimensions belonging to the same feature will be processed by the same FC layer for the purpose of attribution.

VI. FIDELITY METRIC

“Fidelity” is the concept of keeping part of the input and assessing how much information can be retrieved, which is recently formulated to evaluate the explanation methods in NLP [8], [9]. In this section, we propose a fidelity-based metric
Algorithm 1: Fidelity.

**Input:** Original prediction model $F$; Proxy model $Q$; Attribution method $\phi$; (Hyperparameter) Number of selected top-contributed feature dimensions $k$; Training set $T_r$; Testing set $T$

**Output:** $\text{Fidelity}$:

1. $V^k_\phi \leftarrow \{\}; W^k_\phi \leftarrow \{\};$
2. for $X$ in $T_r$ and $T$ do
   3. $\overline{X} \leftarrow$ Preserving the top-$k$ contributed feature-dimensions of $X$ using $\phi$ and $F$, and zero-masking the rest dimensions
5. if $X \in T_r$ then
   6. $V^k_\phi \leftarrow V^k_\phi \cup \{X; F(X)\}$
8. else
   9. $W^k_\phi \leftarrow W^k_\phi \cup \{X; F(X)\}$
10. end if
11. Train $Q$ using $V^k_\phi$
12. Fidelity = $\text{Accuracy}[Q(X) = y | X, y \in W^k_\phi]$

The numbers of layers and attention heads of the sequential input contributed feature dimensions in the input, where $k=100, 10, 5, 1$. Specifically, since our input is time sequential, we average the IG/SG of the input among the time dimension of the input $X$, then choose the top-$k$ dimensions of the averaged IG/SG to be the top-contributed feature dimensions.

B. Prediction Models

1) Long Short-Term Memory (LSTM): We use bidirectional LSTM with two recurrent layers. The probability of dropout is 0.2. The size of the hidden state is 128. After the LSTM, we use a 256-dimensional FC layer and a tanh function to compute the class scores $P(y | X)$.

2) Transformer: The numbers of layers and attention heads are 2 and 8, respectively. We set the dropout probability of the Transformer to be 0.1 and the hidden dimension to be 256. After the Transformer, a 256-dimensional FC layer and tanh function are used to compute $P(y | X)$.

C. Attribution Methods

We compare the “explanatory power” of two attribution methods, IG and SG, on two prediction models (LSTM and Transformer) and four tasks (“Tyrant,” “win,” “kill,” and “be-kill”). For each task, we use the attribution methods to find the top-$k$ contributed feature dimensions in the input, where $k \in \{100, 10, 5, 1\}$. Specifically, since our input is time sequential, we average the IG/SG of the input among the time dimension of the input $X$, then choose the top-$k$ dimensions of the averaged IG/SG to be the top-contributed feature dimensions.

1) Integrated Gradients (IG): We apply (2) to realize IG, and choose the dividing steps to be $\text{steps} = 100$.

2) SmoothGrad (SG): We apply (3) to realize SG with $\text{steps} = 100$, and set the standard derivation $\sigma$ of Gaussian noise for the $i$th dimension of $X$ to be $0.15 \cdot (\max(X_i) - \min(X_i))$.

D. Evaluation Metrics

1) Accuracy: We use the prediction accuracy as the evaluation metric for the aforementioned two prediction models for the four tasks.

2) Fidelity: We evaluate the explanatory power of different pairs of attribution method (IG and SG) and prediction model (LSTM and Transformer) with the Fidelity metric using (4) and Algorithm 1. We conduct these experiments by preserving the top-$k$ contributed feature dimensions, where $k \in \{100, 10, 5, 1\}$.

VII. EXPERIMENTS

A. Experimental Settings

We conduct experiments on the dataset mentioned previously. We set 5000 games from the dataset as the validation set and another 5000 games as the testing set. The rest 40 278 games are the training set.

The sequence-length $l$ of the sequential input $X = [x^{l-1}; x^{l-2}; \ldots; x^0]^T$ is 5.\footnote{Prior experiments showed that using data with sequence length longer than 5 did not bring evident improvement to the accuracy.} For “Tyrant,” “kill,” and “be-kill” tasks, we choose the input data at $S$ seconds intervals before the events’ happening time, where $S \in \{1, 5, 10, 15, 20\}$. For “win” task, we choose the input data every 60 s from the beginning of each game. We also assess the accuracy of the “win” task at different game times, ranging from 40.0 s (the beginning of the games) to 20.0 min. The accuracy of the “win” tasks is accessed averagely (with respect to the predictions at different game times) and separately.

B. Prediction Models

1) LSTM: We use bidirectional LSTM with two recurrent layers. The probability of dropout is 0.2. The size of the hidden state is 128. After the LSTM, we use a 256-dimensional FC layer and a tanh function to compute the class scores $P(y | X)$.

2) Transformer: The numbers of layers and attention heads are 2 and 8, respectively. We set the dropout probability of the Transformer to be 0.1 and the hidden dimension to be 256. After the Transformer, a 256-dimensional FC layer and tanh function are used to compute $P(y | X)$.

C. Attribution Methods

Table I indicates that the two models achieve close accuracy, while Transformer is slightly more accurate. For “Tyrant,” “kill,” and “be-kill” tasks, the prediction accuracy decreases...
when the prediction interval $S$ increases (from 1.0 to 20.0 s), which is logical since it is easier to predict an event in the near future than the one in the distant future. For example, if the Tyrant is killed at game time $t$-seconds, then at $t-5$-s, we are almost sure that the team that has an advantage will seize the Tyrant, while at $t-20$-s, the future situations are not that clear. For the “win” tasks, Table I shows the average prediction accuracy across the different game times.

From Table I, we can also conclude that “Tyrant” prediction is the most accurate one and “win” prediction is the next, while “kill” and “be-kill” predictions are less accurate. The underlying reasons are as follows.

1) “Tyrant” and “win” are binary-classification tasks, while there are ten labels for “kill” and “be-kill.”

2) It is much easier to predict the macroscale events (such as which team will seize the Tyrant and which team will win) than to predict the microscale events (which hero exactly will be killed or kill others) since there is too much uncertainty and variability for microscale events.

From Table I, we can also find that as ten-label classification tasks, for $S = 1$, $S = 5$, and $S = 10$, the “be-kill” tasks are more accurate than the “kill” tasks with the same experimental settings, while for $S = 15$ and $S = 20$ s, the “be-kill” tasks are less accurate. This means that predictions for “be-kill” tasks are more accurate in the near future (within 10 s), while “kill” predictions are more accurate in the distant future (longer than 10 s). A possible explanation for this phenomenon is as follows: In the near future, it is easier to predict who will be killed by checking whose situation is the worst, while there might be several possible candidate killers, making it relatively harder to predict which one of them exactly will be the killer. However, situations will definitely change in the distant future, such as 20 s later. Therefore, we are no longer sure which hero will be in the worst situation then, while it is relatively more accurate to predict which hero will be the potential killer by checking who is the most powerful one.

We further test the accuracy of the two models at different game times for the “win” task to investigate the nature of outcomes for MOBA games. As shown in Fig. 10, the prediction accuracy of both models first increases as the games progress, then declines in the late-game stages (after 12.5-min of game play). This phenomenon happens due to the following reasons.

1) In the early-game stages (before 12.5-min), the games become more predictable as time goes on because the leading team will accumulate its advantages in gold, level, and equipment.

2) In the late-game stages (after 12.5-min), the level and equipment of both teams reach a maximum. Therefore, games are increasingly affected by uncertainty, such as players’ accidental mistakes, and therefore, are harder to predict.

B. Fidelity

Fidelity of different attribution methods (IG and SG) and prediction models (LSTM and Transformer) with respect to $k$ top-contributed feature dimensions ($k \in \{100, 10, 5, 1\}$) is shown in Table II. With a few exceptions, IG and Transformer achieve the highest Fidelity for “win,” “kill,” and “be-kill” tasks. Fidelity decreases when the number of top-contributed feature dimensions $k$ changes from 100 to 1, because when we preserve fewer features, less information of the game can be retrieved.

Experiments show that the Fidelity for “Tyrant” and “win” tasks is higher than the fidelity for “kill” and “be-kill” tasks. One reason is that the first two tasks are binary classification
TABLE III
FIDELITY OF DIFFERENT STEPS VALUES FOR THE “WIN” PREDICTION TASKS OF IG AND SG METHODS AND TRANSFORMER

| Attribution + Model | Top | Fidelity |
|---------------------|-----|----------|
|                     | steps10 | steps50 | steps100 | steps300 | steps500 |
| IG + Transformer    | 100    | 0.950   | 0.952    | 0.956    | 0.954    | 0.951    |
|                     | 10     | 0.977   | 0.889    | 0.890    | 0.883    | 0.881    |
|                     | 5      | 0.852   | 0.867    | 0.866    | 0.863    | 0.870    |
|                     | 1      | 0.792   | 0.805    | 0.799    | 0.796    | 0.792    |
| SG + Transformer    | 100    | 0.925   | 0.930    | 0.936    | 0.933    | 0.930    |
|                     | 10     | 0.875   | 0.860    | 0.870    | 0.884    | 0.875    |
|                     | 5      | 0.834   | 0.838    | 0.837    | 0.837    | 0.832    |
|                     | 1      | 0.622   | 0.622    | 0.622    | 0.622    | 0.622    |

Fig. 11. Top-five features attributed by IG and Transformer when two teams are fighting for Tyrant. Transformer predicts that the red team will seize the Tyrant.

... tasks and the last two tasks are ten-label classification tasks. Therefore, the fidelity (defined by the accuracy of the proxy model) of “Tyrant” and “win” is higher. The other reason is that to predict which team will win or seize the Tyrant, we mainly rely on a small number of critical features (such as gold-difference and Tyrant’s distances to heroes); however, to predict which hero will be the next killer or be-killer one, we need to consider more factors, such as hero-skill information, hero-level, locations, hp, and many other important features.

C. Parameters

We further conduct an additional experiment to evaluate the effect of the choice of steps on the final fidelity result. We assess the fidelity of different steps values of IG and SG for the “win” prediction task with Transformer as the prediction model. From Table III, we can see that there is little change in fidelity for steps ranging in [10,500], which indicates that fidelity is not sensitive to the choice of steps.

D. Case Study

A case study is given in which the two teams are fighting for the Tyrant. As Fig. 11 shows, Transformer predicts that the red team will get the Tyrant with probability 86%, and IG further attributes the prediction result to the following five reasons.

1) Distances between the Tyrant and the heroes: The red team’s heroes are closer to the Tyrant, and therefore, have a better chance of killing the Tyrant.
2) Hero-3 has died: Hero-3 belongs to the blue team, so the blue team has a disadvantage.
3) Gold difference.
4) Kill-count difference: The blue team has a disadvantage in terms of gold difference and kill-count difference.
5) The skill-3 (ultimate skill) of hero-1 is of a low level: This skill is essential for group fighting, so hero-1’s team (blue team) is not capable of seizing the Tyrant.

Eventually, the red team indeed kills the Tyrant.

IX. CONCLUSION

In this article, we attempt to address two main issues for in-game MOBA events: insufficient in-game features and lack of interpretability. We first collect a large-scale HoK dataset containing rich in-game features. To predict four important events (“Tyrant,” “win,” “kill,” and “be-kill”) of HoK in an interpretable manner, we train two sequence modeling networks (LSTM and Transformer) based on the collected dataset and adopt two attribution methods, IG and SG, to give human-interpretable explanations of the prediction results. In addition, a fidelity-based metric is proposed to evaluate the explanatory power of the attribution methods and prediction models. Experiments show that LSTM and Transformer suit well the prediction tasks in terms of accuracy, and IG outperforms SG in terms of the Fidelity metric in our scenarios. With the popularity of MOBA esports, the interpretable event predictions are becoming useful for more and more MOBA-related applications. Our research can act as a first step that could, in our future work, be followed by the development of a user facing tool and a user study that demonstrates and confirms the interpretability.

REFERENCES

[1] K. Conley and D. Perry, “How does he saw me? A recommendation engine for picking heroes in Dota 2,” Np. nd Web, vol. 7, 2013.
[2] Y. Yang, T. Qin, and Y.-H. Lei, “Real-time eSports match result prediction,” 2016, arXiv:1701.03162.
[3] A. Semenov, P. Romov, S. Korolev, D. Yashkov, and K. Neklyudov, “Performance of machine learning algorithms in predicting game outcome from drafts in Dota 2 in Proc. Int. Conf. Anal. Images, Social Netw. Texts, 2016, pp. 26–37.
[4] S. Hochreiter and J. Schmidhuber, “Long short-term memory,” Neural Comput., vol. 9, no. 8, pp. 1735–1780, 1997.
[5] A. Vaswani et al., “Attention is all you need,” in Proc. Adv. Neural Inf. Process. Syst., 2017, pp. 5998–6008.
[6] M. Sundararajan, A. Taly, and Q. Yan, “Axiomatic attribution for deep networks,” in Proc. Int. Conf. Mach. Learn., 2017, pp. 3319–3328.
[7] D. Smilkov, N. Thorat, B. Kim, F. Viégas, and M. Wattenberg, “Smoothgrad: Removing noise by adding noise,” 2017, arXiv:1706.03825.
[8] J. Li, L. Liu, H. Li, G. Li, G. Huang, and S. Shi, “Evaluating explanation methods for neural machine translation,” in Proc. 58th Annu. Meeting Assoc. Comput. Linguistics, 2020, pp. 365–375.
[9] A. Jacovi and Y. Goldberg, “Towards faithfully interpretable NLP systems: How should we define and evaluate faithfulness?,” in Proc. 58th Annu. Meeting Assoc. Comput. Linguistics, 2020, pp. 4198–4205.
[10] K. Kalyanaraman, “To win or not to win? A prediction model to determine the outcome of a Dota2 match,” Univ. California San Diego, CA, USA, Tech. Rep., 2014.
[11] N. Kinkade, L. Jolla, and K. Lim, “Dota 2 win prediction,” Univ. California San Diego, CA, USA, Tech. Rep., pp. 1–13, 2015.
[12] K. Song, T. Zhang, and C. Ma, “Predicting the winning side of Dota2,” School Inf. Library Sci., 2015.
[13] I. Makarov, D. Savostyanov, B. Litvyakov, and D. I. Ignatov, “Predicting winning team and probabilistic ratings in Dota 2 and “counter-strike: Global offensive” video games,” in Proc. Int. Conf. Anal. Images, Social Netw. Texts, 2017, pp. 183–196.
[14] N. Wang, L. Li, L. Xiao, G. Yang, and Y. Zhou, “Outcome prediction of Dota2 using machine learning methods,” in Proc. Int. Conf. Math. Artif. Intell., 2018, pp. 61–67.
[15] V. J. Hodge, S. M. Devlin, N. J. Sephton, F. O. Block, P. I. Cowling, and A. Drachen, “Win prediction in multi-player eSport: Live professional match prediction,” IEEE Trans. Games, vol. 13, no. 4, pp. 368–379, Dec. 2021.

[16] Z. Yang et al., “Interpretable real-time win prediction for Honor of Kings, a popular mobile MOBA Esport,” IEEE Trans. Games, 2020.

[17] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” in Proc. 31st Int. Conf. Neural Inf. Process. Syst., 2017, pp. 4768–4777.

[18] P. Sun et al., “TStarBots: Defeating the cheating level builtin AI in Starcraft II in the full game,” 2018, arXiv:1809.07193.

[19] D. Ye et al., “Mastering complex control in MOBA games with deep reinforcement learning,” in Proc. AAAI Conf. Artif. Intell., 2020, pp. 6672–6679.

[20] Z. Zhang et al., “Hierarchical reinforcement learning for multi-agent MOBA game,” 2019, arXiv:1901.08004.

[21] OpenAI, “OpenAI five,” 2018. [Online]. Available: https://blog.openai.com/openai-five/

[22] A. Smerdov, B. Zhou, P. Lukowicz, and A. Somov, “Collection and validation of psychophysiological data from professional and amateur players: A multimodal eSports dataset,” 2020, arXiv:2011.00958.

[23] A. Smerdov, A. Somov, E. Burnaev, B. Zhou, and P. Lukowicz, “Detecting video game player burnout with the use of sensor data and machine learning,” IEEE Internet Things J., vol. 8, no. 22, pp. 16680–16691, 2021.

[24] A. Katona et al., “Time to die: Death prediction in Dota 2 using deep learning,” in Proc. IEEE Conf. Games, 2019, pp. 1–8.

[25] S. He, Z. Tu, X. Wang, L. Wang, M. R. Lyu, and S. Shi, “Towards understanding neural machine translation with word importance,” in Proc. Conf. Empirical Methods Natural Lang. Process. 9th Int. Joint Conf. Natural Lang. Process. (EMNLP-IJCNLP), 2019, pp. 953–962.

[26] C. Chen, O. Li, D. Tao, A. Barnett, C. Rudin, and J. K. Su, “This looks like that: Deep learning for interpretable image recognition,” in Proc. Adv. Neural Inf. Process. Syst., 2019, pp. 8930–8941.

[27] C. Buciluă, R. Caruana, and A. Niculescu-Mizil, “Model compression,” in Proc. 12th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining, 2006, pp. 535–541.

[28] A. Polino, R. Pascuana, and D. Alistarh, “Model compression via distillation and quantization,” 2018, arXiv:1802.05668.

[29] G. Hinton, O. Vinyals, and J. Dean, “Distilling the knowledge in a neural network,” Stat, vol. 1050, p. 9, 2015.

[30] Y. Liu, W. Che, H. Zhao, B. Qin, and T. Liu, “Distilling knowledge for search-based structured prediction,” in Proc. 56th Annu. Meeting Assoc. Comput. Linguistics (Volume 1: Long Papers), 2018, pp. 1393–1402.

[31] A. Adadi and M. Berrada, “Peeking inside the black-box: A survey on explainable artificial intelligence (XAI),” IEEE Access, vol. 6, pp. 52138–52160, 2018.

[32] D. Gunning, M. Stefl, J. Choi, T. Miller, S. Stumpf, and G.-Z. Yang, “XAI-explainable artificial intelligence,” Sci. Robot., vol. 4, no. 37, 2019, Art. no. eaay7120.

[33] F. K. Došišović, M. Brčić, and N. Hlupić, “Explainable artificial intelligence: A survey,” in Proc. 41st Int. Conf. Inf. Commun. Technol., Electron. Microelectron., 2018, pp. 0210–0215.

[34] W. Samek and K.-R. Müller, “Towards explainable artificial intelligence,” in Explainable AI: Interpreting, Explaining and Visualizing Deep Learning, Switzerland: Springer, 2019, pp. 5–22.

[35] A. B. Arrieta et al., “Explainable artificial intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI,” Inf. Fusion, vol. 58, pp. 82–115, 2020.

[36] A. Das and P. Rad, “Opportunities and challenges in explainable artificial intelligence (XAI): A survey,” 2020, arXiv:2006.11371.

[37] G. Vilone and L. Longo, “Explainable artificial intelligence: A systematic review,” 2020, arXiv:2006.00093.

[38] P. P. Angelov, E. A. Soares, R. Jiang, N. I. Arnold, and P. M. Atkinson, “Explainable artificial intelligence: An analytical review,” Wiley Interdiscipl. Rev.: Data Mining Knowl. Discov., vol. 11, no. 5, 2021, Art. no. e1424.

[39] L. Longo, R. Goebel, F. Lecue, P. Kieseberg, and A. Holzinger, “Explainable artificial intelligence: Concepts, applications, research challenges and visions,” in Proc. Int. Cross-Domain Conf. Mach. Learn. Knowl. Extraction, 2020, pp. 1–16.