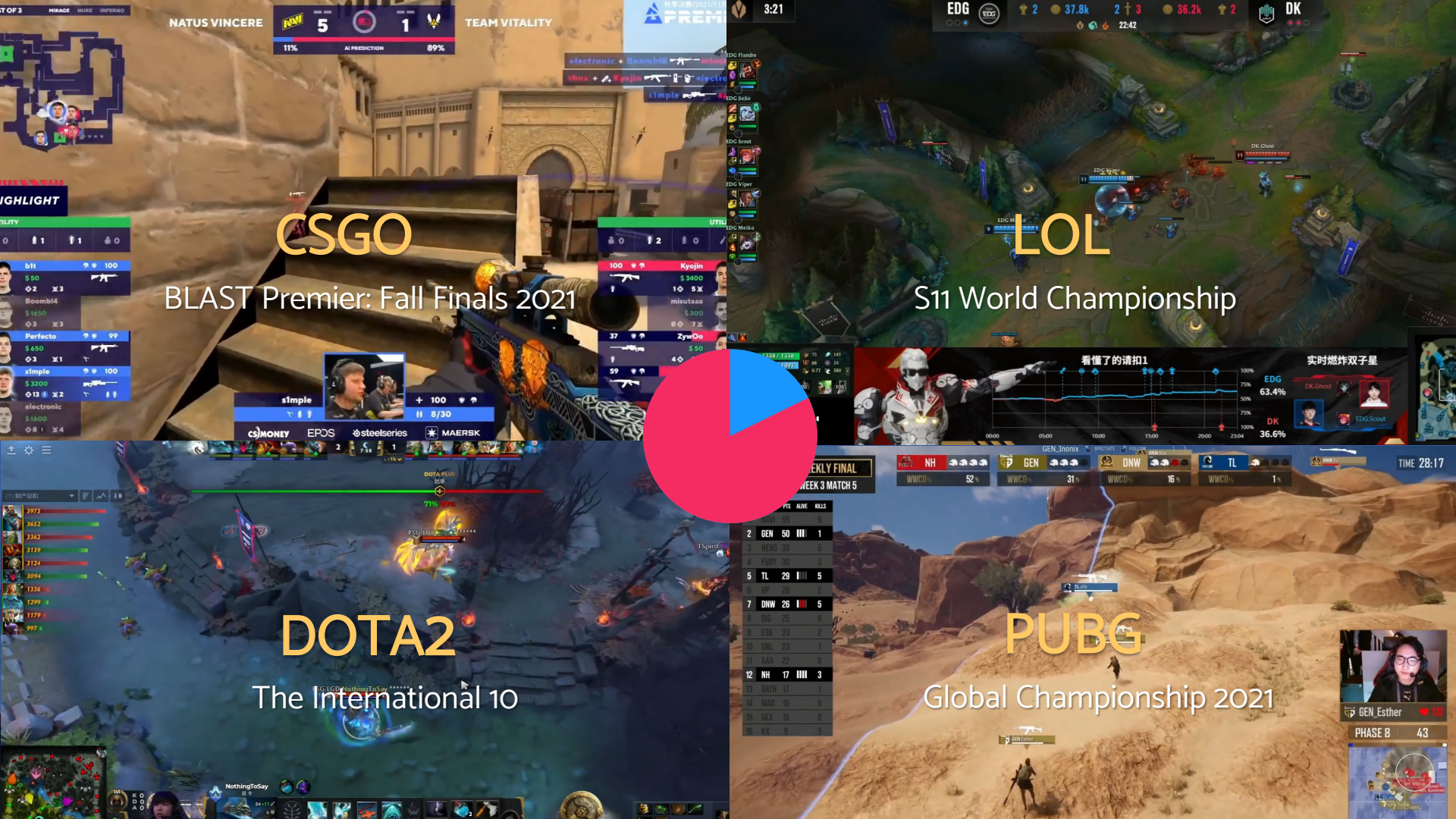
Machine Learning Based ESports Winner Prediction

ShanghaiTech CS181 Project : LOL Esports Prediction

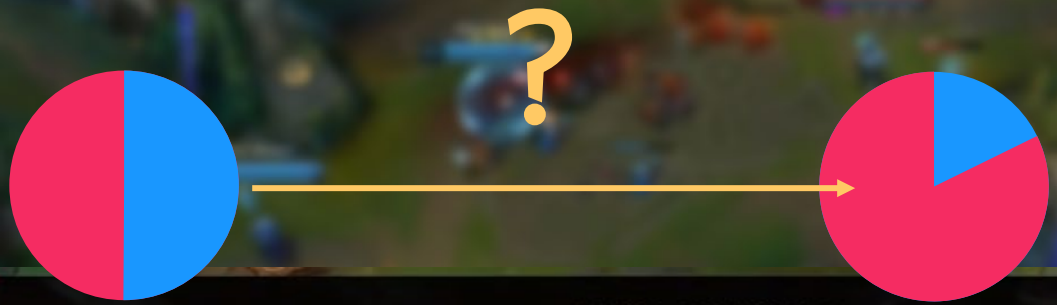
连奕航/邬一闻/熊昕洋/吕钧霆/余诗博


01
Introduction
When watching esports matches,
did you notice this?


CSGO
BLAST Premier: Fall Finals 2021
LOL
S11 World Championship
DOTA2
The International 10
PUBG
Global Championship 2021

LOL Esports Winner Prediction

- Example: KI Colonel (KFC-AI) Realtime Win Rate
- Who to win the game?



02

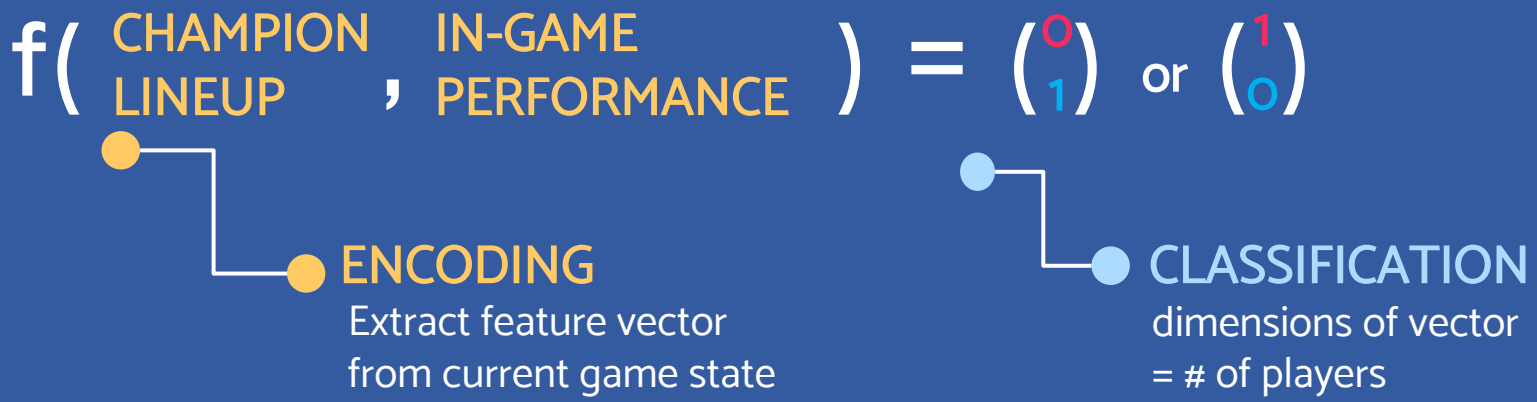
Our Methods

How to determine who can win the game?

WHO WILL WIN THE GAME?

WHO WILL WIN THE GAME?

Supervised Machine Learning Approach



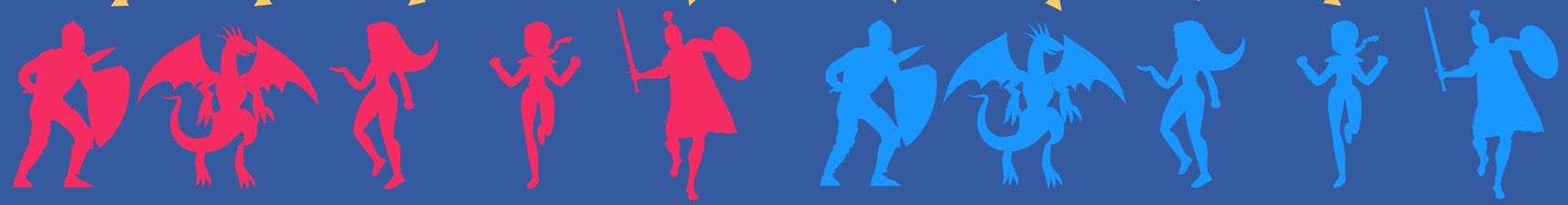
Champion Lineup: Naive Bayes

Champion Lineup: Naive Bayes

Laplace Smoothing → P(Y) Y = { Red, Blue }

→ $P(Y, F_1, F_2, \cdots, Fn)$
$= P(Y) \prod_i P(Fi|Y)$

→ $P(Y|F_1, F_2, \cdots, Fn)$



TOP JUNGLE MIDDLE ADC SUPPORT TOP JUNGLE MIDDLE ADC SUPPORT

$P(F_i|Y)$ i=1, 2, ., 10

Champion Lineup: Naive Bayes

Result & Thoughts

- Result:
  - 54.0% on 2021 pro matches dataset
  - 52.7% on 2015-2018 pro matches dataset
  - even monkey can get 50% accuracy. . .
- Why poor performance?
  - Naive Bayes assumes conditional independence on champions
    - however, champion cooperation&counter matters!
  - Pro players might be influenced little by champion lineup
    - they tend to pick OP champion lineup as possible
  - Game patch differs

Champion Lineup: Logistic Regression

Result & Thoughts

- Using Gradient Descent to optimize log loss with sigmoid probability
- Result:
  - 54.1% on 2021 pro matches dataset
  - 53.2% on 2015-2018 pro matches dataset
- This can prove champion lineup less matters to pros than expected

In-Game Performance

In-Game Performance: Logistic Regression

Result & Thoughts

- Similarly, using Gradient Descent to optimize loss
- Features captured:
  - Difference between blue and red in . . .
  - . . . gold earned/enemies killed/towers destroyed/monsters killed
  - At certain timestep (early stage), like t=15 minutes after beginning
- Result:
  - 76.0% on 2021 pro matches dataset
  - 72.3% on 2015-2018 pro matches dataset

In-Game Performance: RNN

Realtime prediction based on Recurrent Neural Network

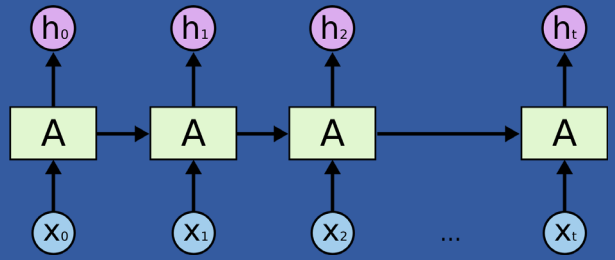
$$X_t = (\ \text{Gold}\ ,\ \text{Tower}\ ,\ \text{Kill}\ ,\ \text{Dragon}\ )_t \qquad t=1,2,\cdots,T$$

- We want to implement real time prediction!
- Features at each previous time step are captured X1, X2, . . . , Xt
- Our choice - RNN !
  - can encode arbitrary-length input (T=5, 10, 15, 20, 25, 30 in our project)
  - can do classification through an additional output layer
  - With LSTM cells, can preserve more information in early time steps
  - LSTM: Long Short Term Memory networks

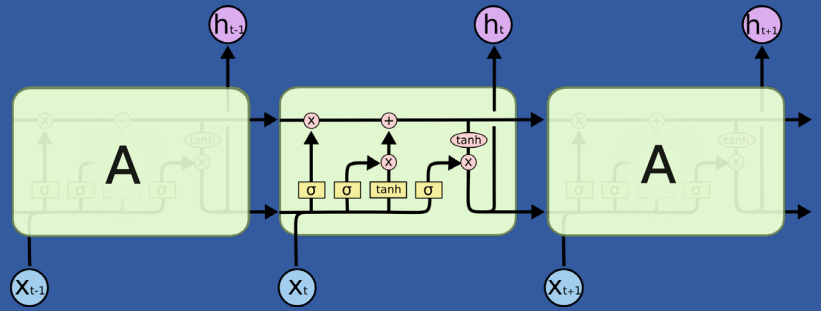
In-Game Performance: RNN & LSTM

Using Toolkit: PyTorch

$X_t = ($ Gold , Tower , Kill , Dragon $)_t$ $t=1,2,\cdots,T$



nn.RNN



nn.LSTM

batch size = 32, learning_rate = 0.001, loss_func = CrossEntropy, activation = ReLU, RNN hidden size = 256, LSTM num layers = 1, train:valid:test split ratio = 6:2:2, early stopping on valid accuracy

In-Game Performance: RNN

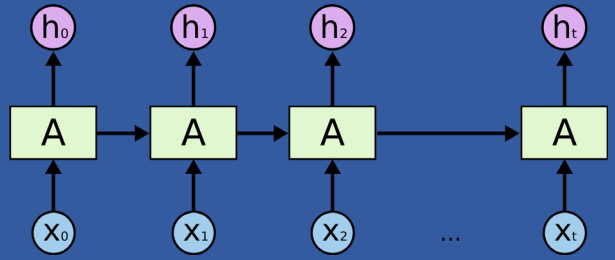
Realtime prediction based on Recurrent Neural Network

- Result on 2015-2018 pro matches dataset :

RNN		LSTM
58.0%	05 min	58.9%
66.5%	10 min	66.7%
73.1%	15 min	73.9%
77.3%	20 min	77.4%
82.7%	25 min	82.9%
84.4%	30 min	85.0%

In-Game Performance: RNN & LSTM

Using Toolkit: PyTorch

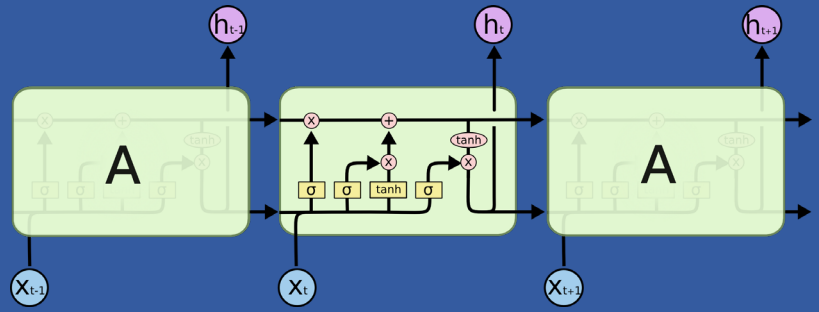


nn.RNN

84.4%

30 min

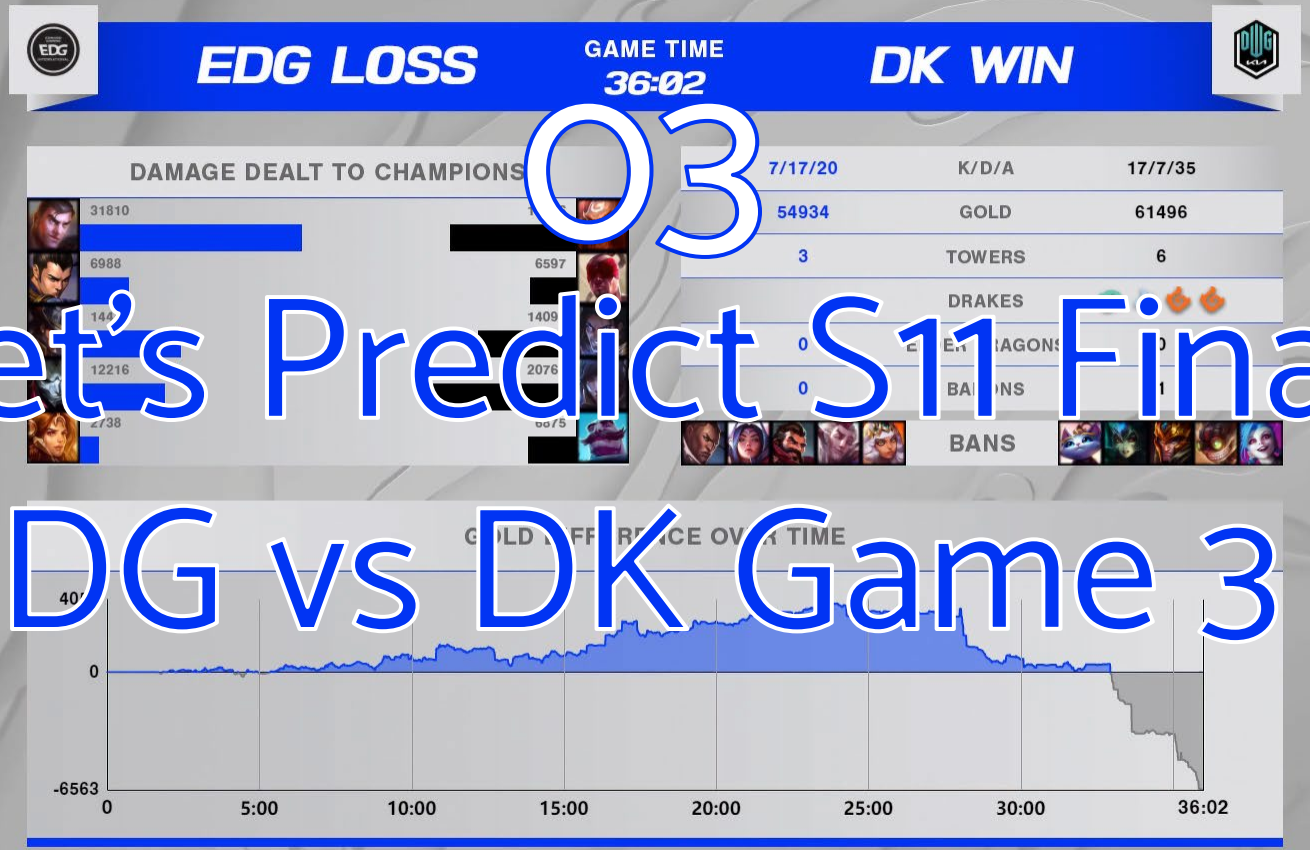
RNN LSTM



nn.LSTM

85.0%

- Improvement of LSTM seems not obvious from accuracy metric
- But we can demonstrate their difference by following example



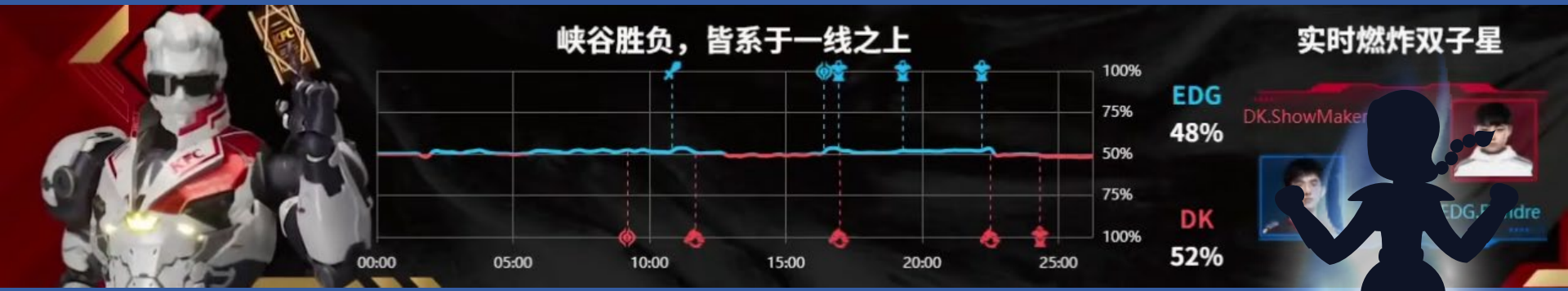
03

Let's Predict S11 Final

EDG vs DK Game 3 !

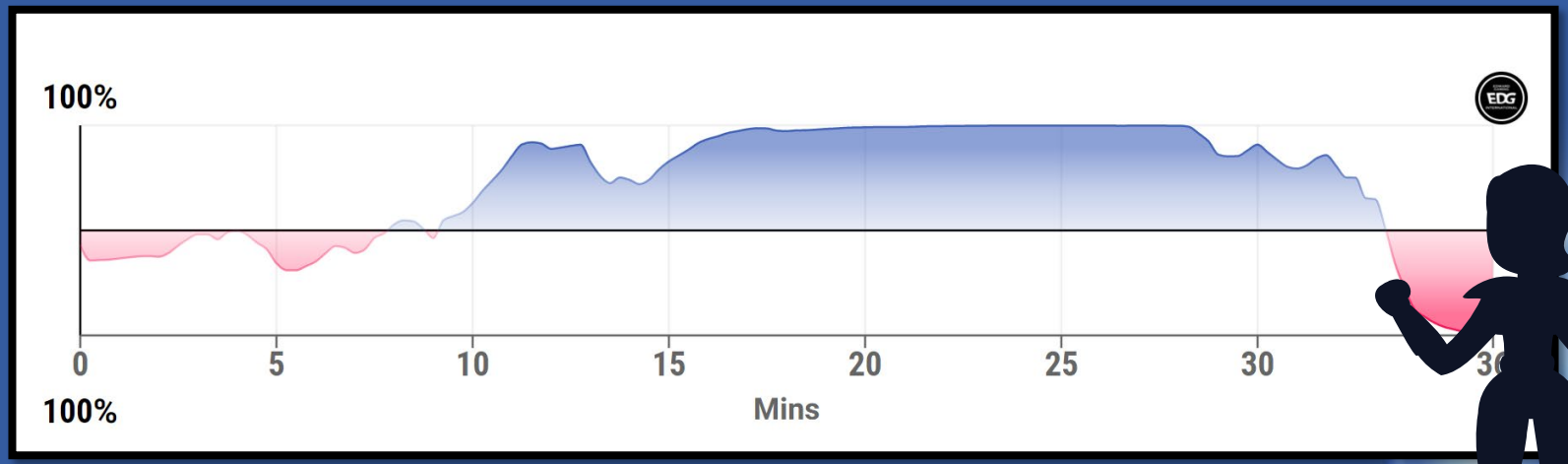
Predict S11 EDG vs DK Game 3

Source: KI Colonel (KFC-AI)



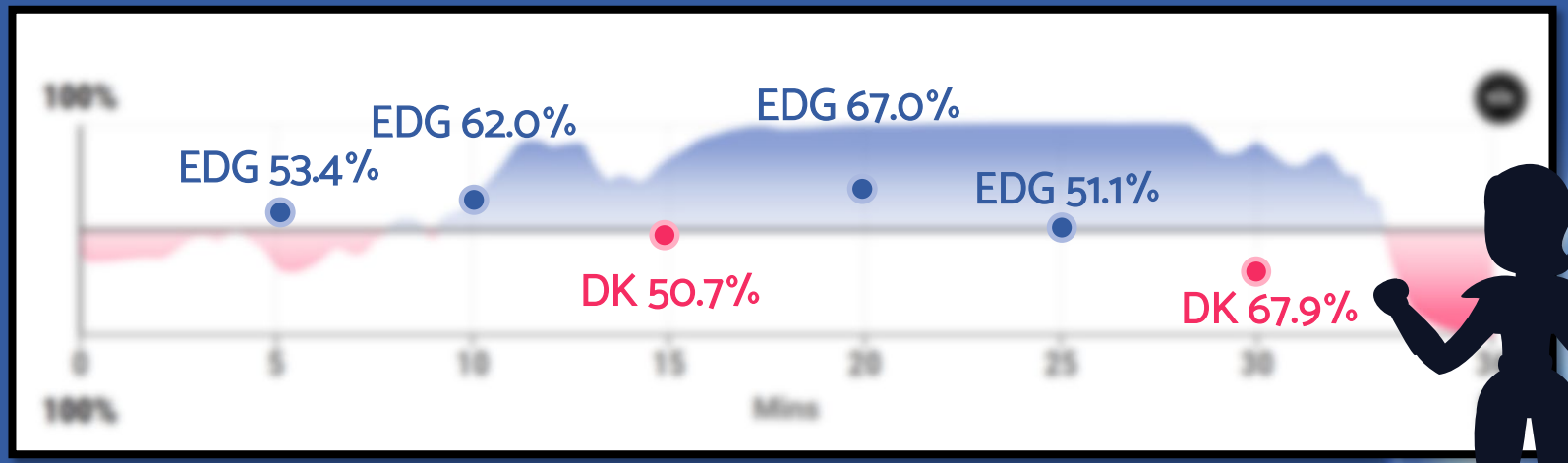
Predict S11 EDG vs DK Game 3

Source: FACTOR.GG



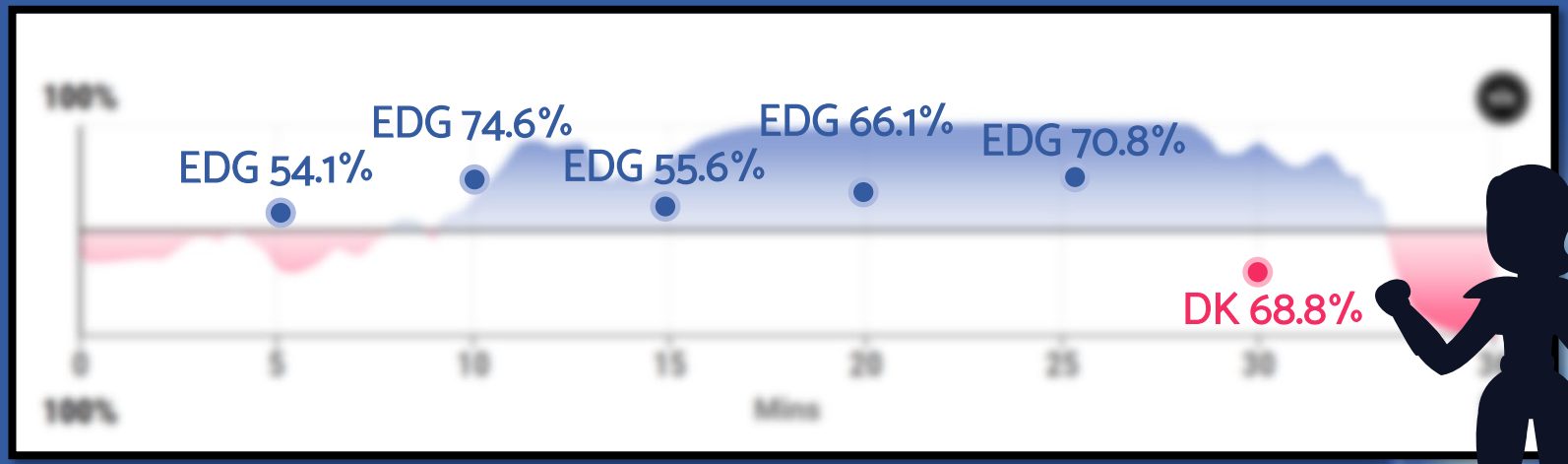
Predict S11 EDG vs DK Game 3

Our LSTM RNN Model Prediction!



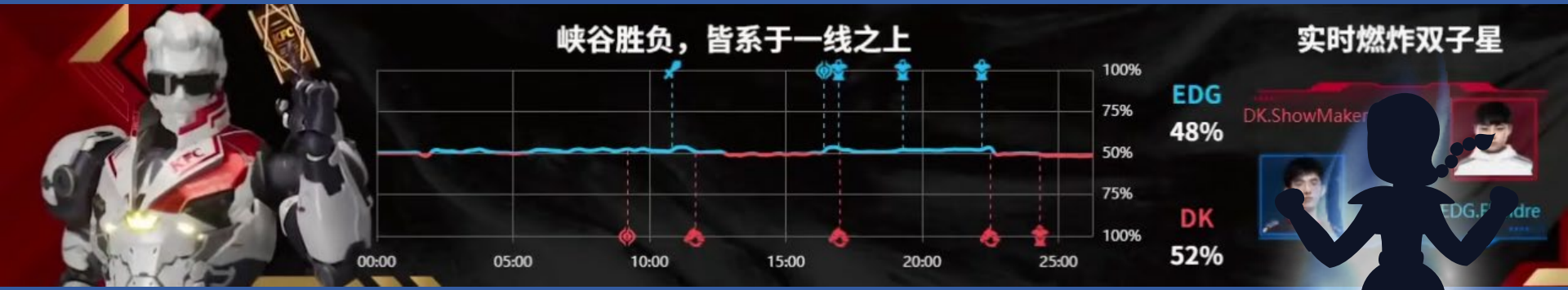
Predict S11 EDG vs DK Game 3

Our RNN Model Prediction!



Predict S11 EDG vs DK Game 3

Source: KI Colonel (KFC-AI)



Predict S11 EDG vs DK Game 3

Our LSTM RNN Model Prediction!

THANKS!