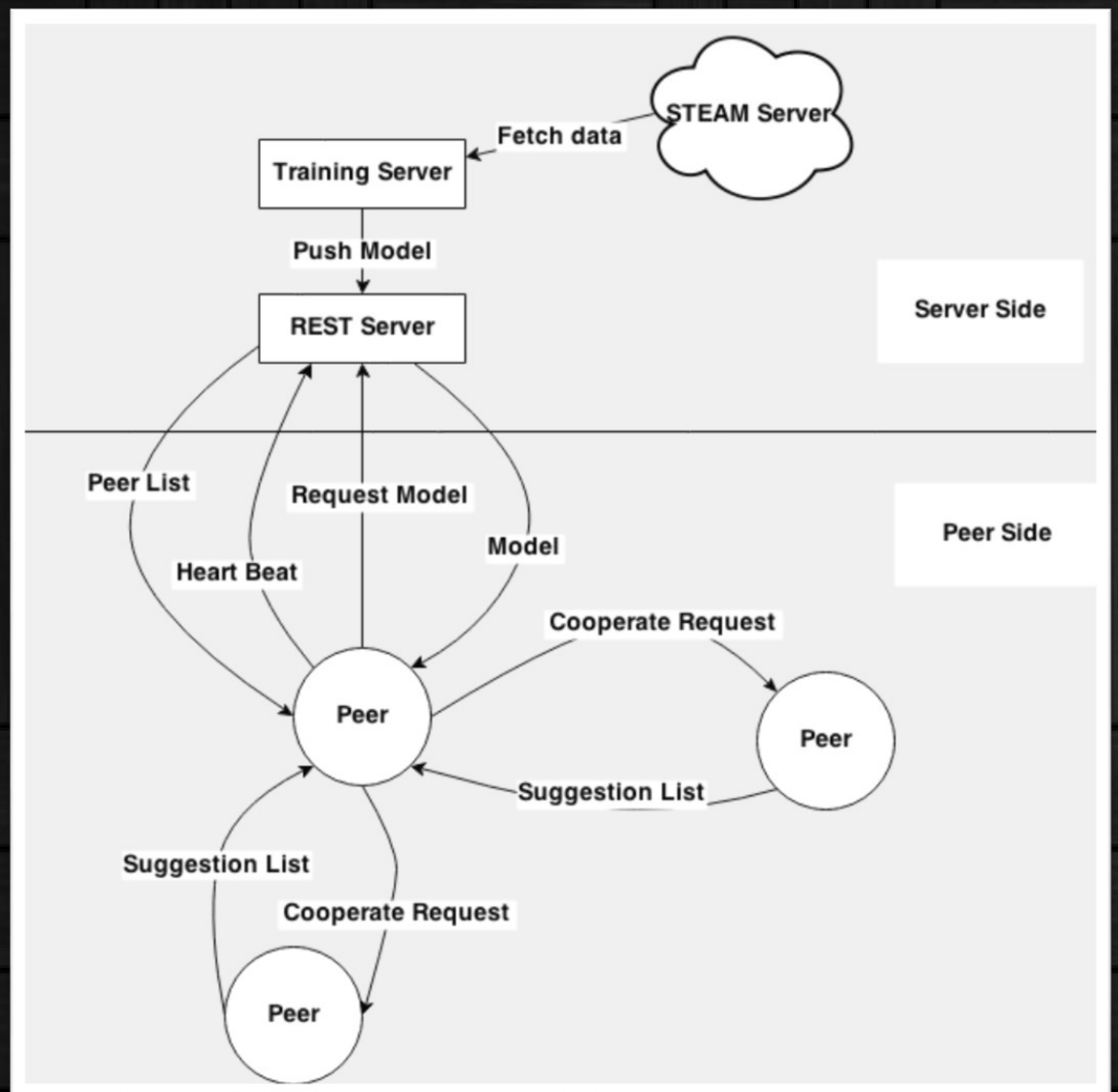
Pro Dota 2 Real Time Strategy Coach

Shy-Yauer Lin, Ruisheng Shi and Amanda Jia

Introduction

- **Dota 2, a multiplayer online battle arena (MOBA) game with more than 12.9 million unique users this April.**
- **The International 6 had a total prize pool of 20.8 millions US dollars, top the e-sports prize pool history. In fact, only Wimbledon($41 million) and Champion League have higher prize pool than Dota.**
- **Counter-hero picking is highly correlated to winning rate.**
- **Our unique selling propositions is that Pro Dota2 is the first real-time, low latency, interactive heroes recommendations engine to help players to maximum their winning rate.**

System Design



- There are two parts:
  - Server side for training learning model and maintaining peers information
  - Peer side for cooperatively recommending hero list
- **Training Server** is a process that
  - Crawl match data from STEAM server, the game company server
  - Filter out invalid matches
  - Train a new model in weekly basis
- **Restful Server** is an Tornado Webserver used to:
  - Handle users' *heart beat message*:
    - Save user information (IP address, port number, and time-stamp) into in-memory database
    - Return a list of peers to user
  - Handle users' *model request message*:
    - Return the classifier to user according to user request.
  - Maintain active user list:
    - Delete user entries that are expired periodically
  - Design Choice:
    - Tornado Webserver: high I/O throughput, moderate CPU efficiency
    - Stateless: low overhead, scalable
    - In-memory database: high throughput

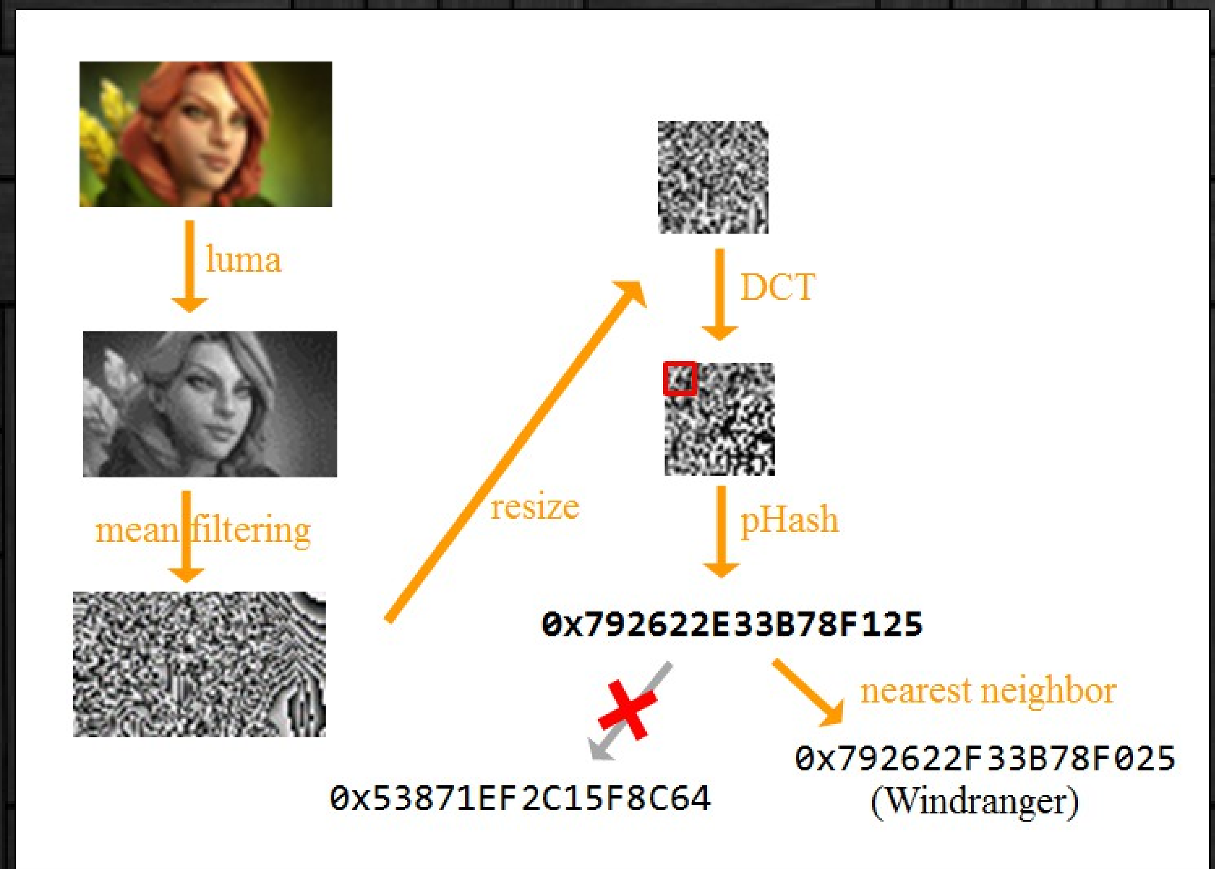
Peer Side

- The predictor process is responsible for:
  - Sends heart beat message to RESTful server
  - Fetches model & peers informations from RESTful server
  - Accept prediction request from UI component
  - Distributes the prediction tasks across a set of peers
  - Aggregate collectively predictions and make final suggestions
  - Responses result to UI component

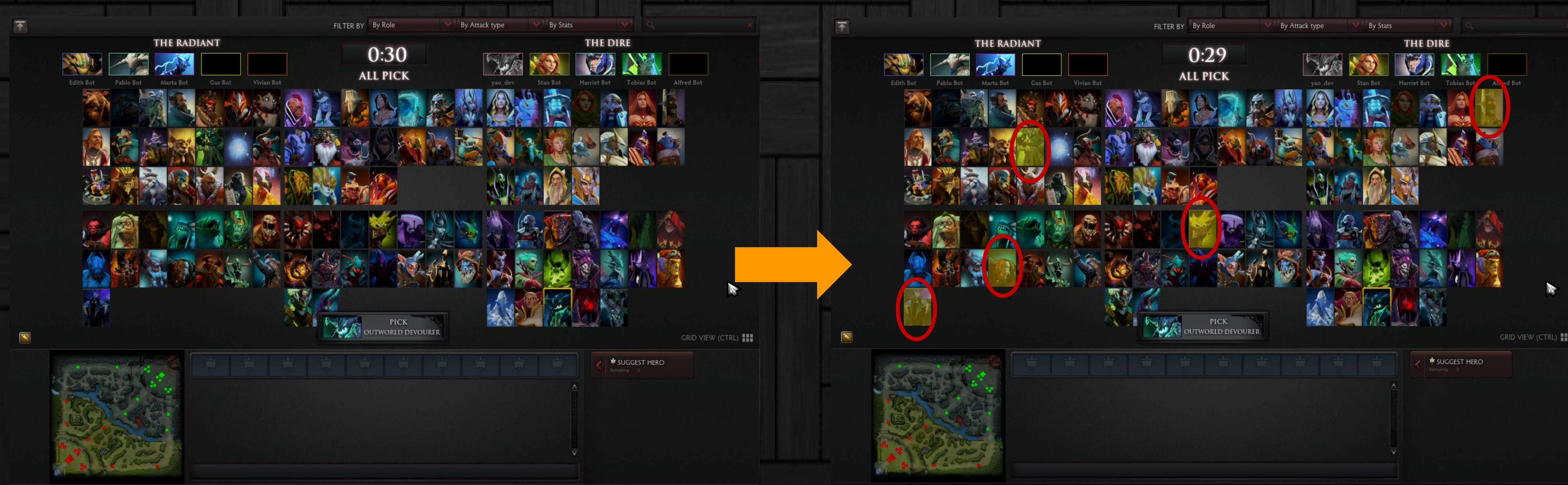
UI Component

Our UI component detects chosen heroes automatically in a blink

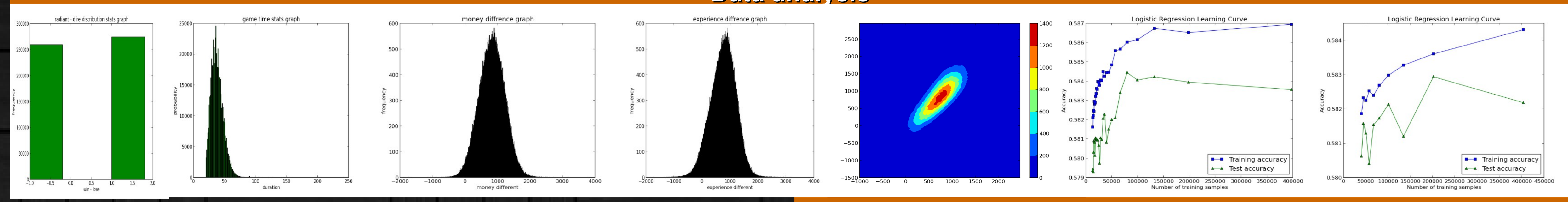
- Fast computer vision technique for heroes' portraits detection
  - System level screenshot
  - Image luma extraction
  - Mean filtering
  - DCT type II conversion
  - Perceptual Hashing to calculate image feature
  - Nearest neighbor in Hamming distance for hero identity decision



In-game Real-time Recommendation



Data analysis



- prediction rate drops 8 percentage vs. Sequential crawled data.
- 1) **Outlier: professional players.** By filtering out 50 (~0.6% of all players)professional players, the prediction score ups by 0.27%(Refer to the right graphs) .
- 2) **Professional and Pro players have much larger proficient heroes pools.** The roughly estimation game hours per week is 22.1 hours per week. For 50 professional players, the time per week is 41.3 hours per week.
- 3) **predictor will favor the heroes with low learning curves instead of the most fit heroes based on heroes unique skills**. This is verified by data from DotaBuff.com, the top 5 highest win rate heroes all have very smooth learning curves. In particular, the top one wraith king is well known recommended hero for newbie

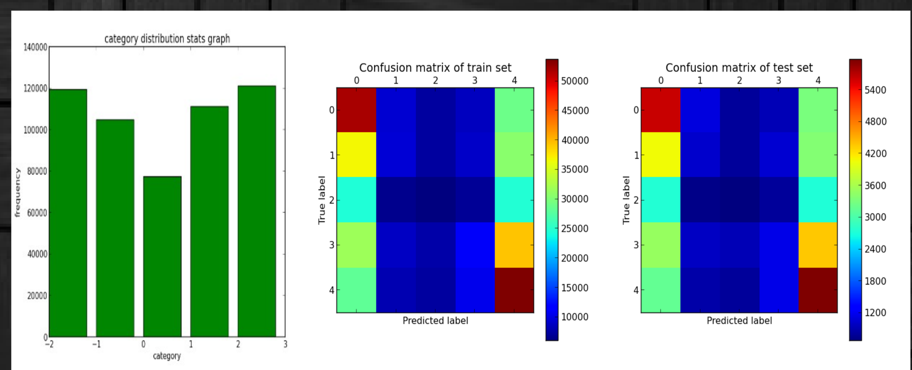
Win-lose prediction accuracy and time

- 397431 matches excluded 50 professional players. score of ensemble decision tree (40 decision trees)is 0.5796. The dumped classifier size is 275.8 MB, the prediction time per query for 44158 queries is 0.18ms
- Linear SVM dumped classifier size is 34.9MB, prediction time is 1.33ms
- Tested on Intel i7-3930K, 3.2Hz * 12 CPUs 16G memory

Score metrics

- Analysis **winner.gold+ experience - loser.gold+experience.** In 97.8% matches, winner has more gold and experience than losers. In particular, this is actually a bi-variable normal distribution.
- we split the matches into five categories based on winner.gold+exp – loser.gold+exp. Then applied a oneVsAll (estimator is linear SVM)

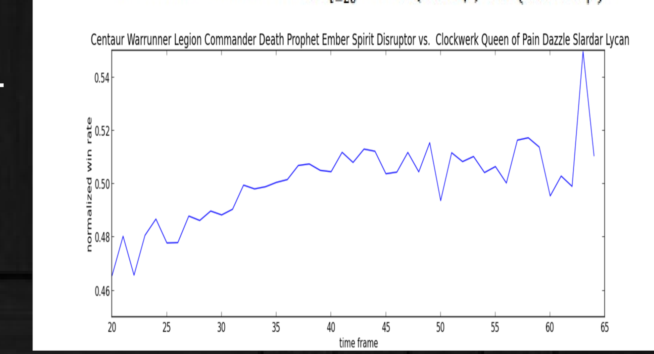
Accuracy is 0.59 for training data. 0.587 for test data.



The model behind the unique selling points

- Probabilistic model predict what heroes a player may choose. The model we are using is a*now + (1-a)rest of all heroes vectors. We set a = 0.05. if you play axe only once 100 matches ago. The probability it will affect is 0.03%. The recall rate is 0.4084 for 5% threshold and 0.6393 for 1%

$$metric(Allies, Enemies) \equiv \sum_{t=20}^{\infty} \Pr(t)\frac{\Pr(Allies|t) - \Pr(Enemies|t)}{\Pr(Allies|t) + \Pr(Enemies|t)}$$



Current competitors & our strength

- DotaBuff provide offline, non-real time statistic result and very naïve processing.
- **our system will provide four revolutionary recommendations**:
- During the loading screen, our system will trigger the vision recognition and record all players' user names and return a list of most recently used heroes and more importantly, **a probabilistic model of the heroes enemy will select.**
- **Propose the fittest position for each player**
- During the heroes picking process, our system should interactive **recommend the heroes combinations**. low latency very important
- After hero selection stage, we provide user a chart of time related win rate estimation.
- Expertise Knowledge: Ruisheng has played dota since 2007, He know some professional players who can direct promote our product in the near future in Twitch(10,000 average live watchers)

Crawling method and Data Format

- Different skill set of players has total different selection and preference of heroes. Take one hero, earth spirit as an example, earth spirit has an overall win rate 33.32%. It is the lowest win rate hero, 5% away from second lowest win rate heroes. But he is actually the most 'imba' hero in Pro skill players in last patch.
- We first manually find 525 players from the world leaderboards, then graph walk to explore the players they met during the play and apply a filtering. Then we extract top 7315 players and analysis 546, 097 matches from these players.
- We separate the players into ten skill levels. But We have no way to know other skill level player's statistic. Therefore we can only release our product first to collect data from our client. Then we can train separate classifier for other levels