Significantly Improving your Skill System with TrueSkill® Through Time

Dr. Josh Menke

Lead Engagement Designer

343 Industries

Outline

I. Skill Rating System Review
II. Common Extensions and their Problems
III. TrueSkill® Through Time

What is skill?

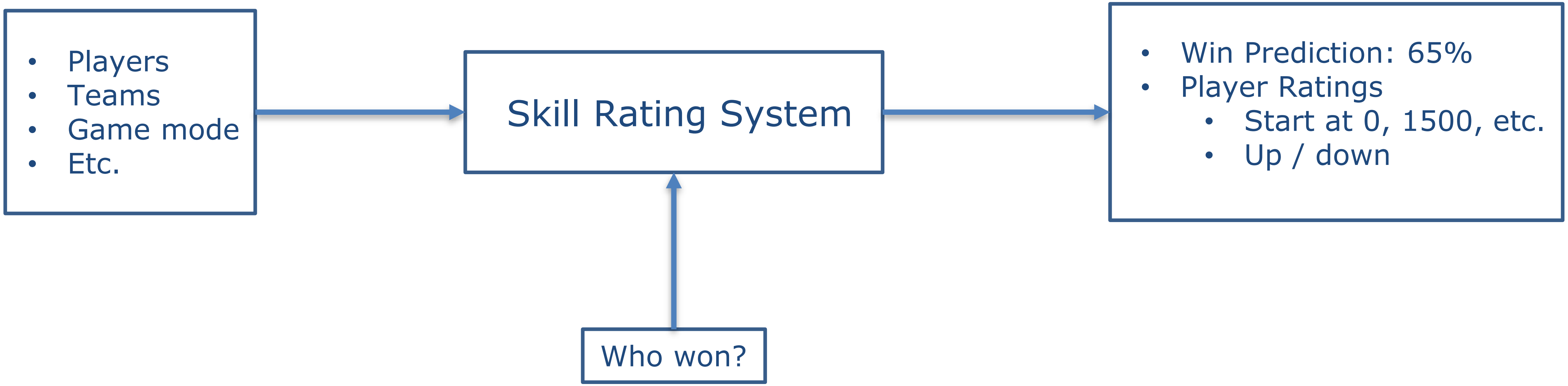
- Dictionary:

  The ability to do something well

- For this talk:

  The ability to do well at consistently winning matches

What is a skill system? Matches to Ratings



- Players
- Teams
- Game mode
- Etc.

Skill Rating System

Who won?

- Win Prediction: 65%
- Player Ratings
  - Start at 0, 1500, etc.
  - Up / down

Popular Skill Rating Systems

- **Elo**
  - Pioneering work, probably most popular
  - Requires more matches to converge, requires tight matchmaking
- **Glicko**
  - Requires less matches to converge than Elo, doesn’t require tight MM
  - not naturally adapted to teams or draws
- **TrueSkill**
  - Requires even less matches than both Elo and Glicko to converge
  - Designed for teams and draws

Good Skill Rating Systems

1. **Accurate:** Higher-skilled wins more often
2. **Fast:** How many matches? Win% of a new player?
3. **Extensible:** Can it handle needed extensions?

Extension Evaluation Process

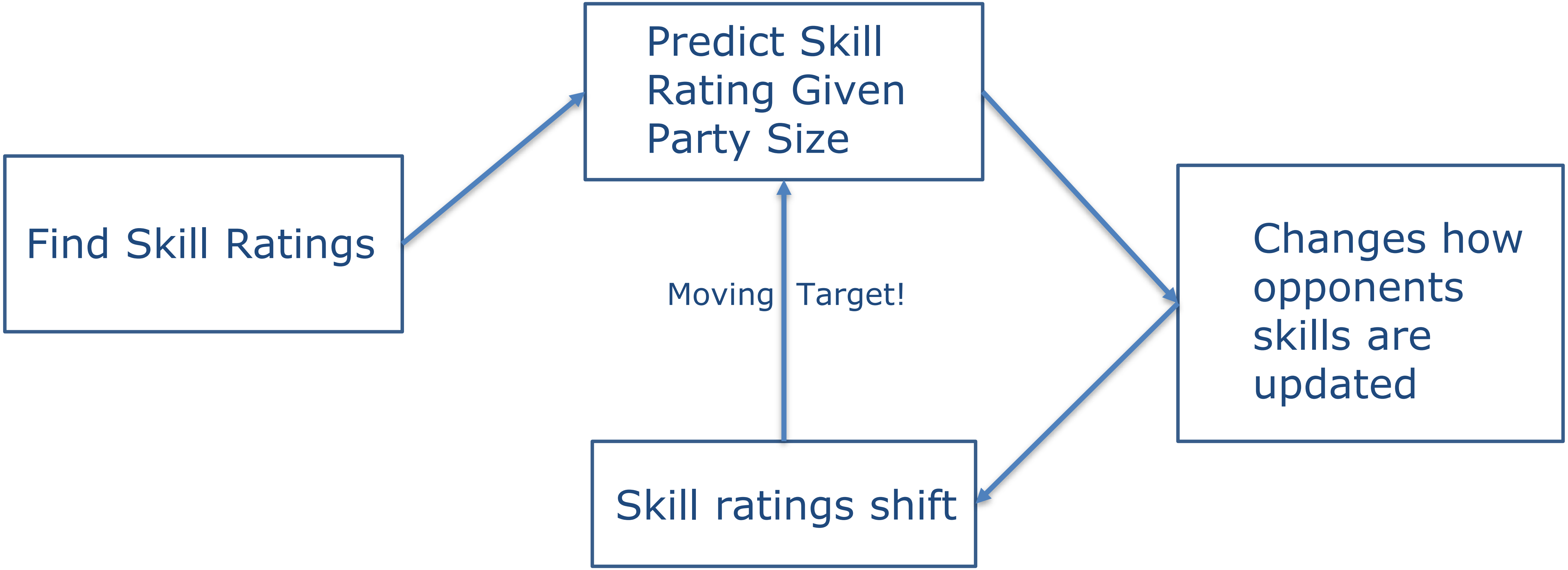
1. Identify a shortcoming of the skill system.
   - E.g.: Playing in premade parties isn’t handled.

2. Consider what data could help improve the shortcoming
   - E.g.: Are they in a premade party? What size of party? Etc.

3. Verify the data is relevant *before* implementing.
   - E.g.: Do premade parties actually have a higher win% than predicted?

4. Decide the best way to incorporate the new data
   - E.g.: Change the skill rating based on the party size.

Moving Target Problem



Find Skill Ratings

Predict Skill Rating Given Party Size

Changes how opponents skills are updated

Skill ratings shift

Moving Target!

Game Modes

1. Shortcoming: Players have different skills per mode, class, platform
   - Motivations: Ranking, Cross-play, not afraid to try new modes, classes, etc.

2. Data: Set of players who each play multiple modes

3. Verify:
   - Win % lower than predicted between modes
   - OR win % lower than predicted for the first game on a new mode

4. Implementation: Have a separate rating per mode
   - Shortcoming: need more matches to converge if not sharing between
   - Moving target problem if sharing is done with an external model

Party Size

1. Shortcoming: Players perform better in parties
   - OR players get defeated unfairly by parties
   - Games limit party size, or restrict MM based on it

2. Data: Matches with party sizes and who won

3. Verify: Win% higher than predicted in larger parties

Party Size Example

Party Size	Prediction %	Win %
1	49	49
2	50	50
3	49	48
4	**53**	**58**

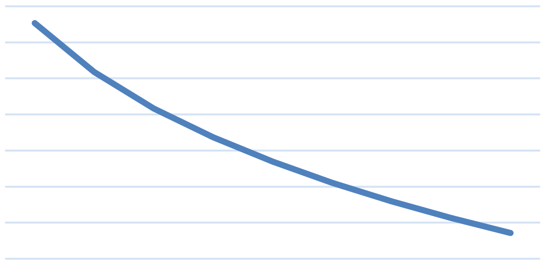
Party Size: 4. Implementation

- Fit external model to learned skill ratings to find party advantages
  - Forces external changes to skill ratings: **Moving Target Problem**
- OR: add a party size offset to the skill system as an extra player per party
  - Have to update a global extra player after match: tricky to engineer (contention)
- Separate ratings for every party or party size
  - More parameters, per game mode, grows fast, requires more matches to converge
  - Throws out known base skill of the player

New Players

1. Shortcoming: new players are worse than average

New player drop-off:



1. Data: win% given # of games a player has played

2. Verify: New players win less than expected

New Players 3: Verifying

Games Played	Predicted win%	Actual win%
(first game)	**49**	**44**
1	**49**	**45**
2	**49**	**46**
3	**49**	**47**
4	**49**	**47**
5	**49**	**47**
6	49	48
7	49	48
8	50	48
≥9	50	51

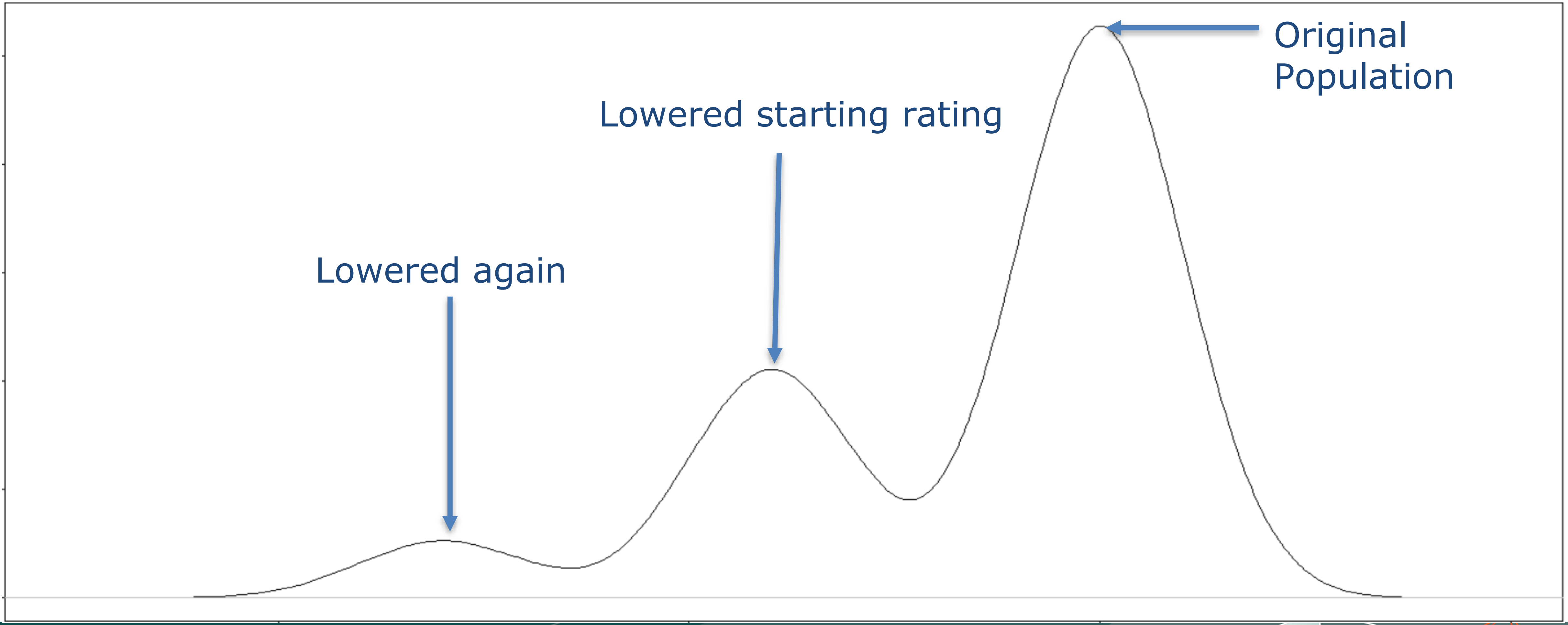
New Players: 4. Implementation

- Need to match against lower-skilled opponents

- Matchmake them lower without changing skill rating
  - How much lower? Find in the data.
  - How fast should you move them back up? **Not linear**. Per mode.
  - Wrong skill update for the opponents

- Start new players with a lower skill rating to fix that
  - Bad **moving target problem**
  - Shifts population down as you go

Moving Target with New Players



Kills, Deaths, Spend, XP, Mana …

1. Shortcoming: Should use post-match metrics like kills
   1. For Ranking: Due recognition in team games
   2. For Matchmaking: Smurfs placed faster

2. Data: The stat in question, per player, per match

3. Verify:
   - Can't compare stats in current game to win% (cheating)
   - Compare previous game or pre-game average to win%

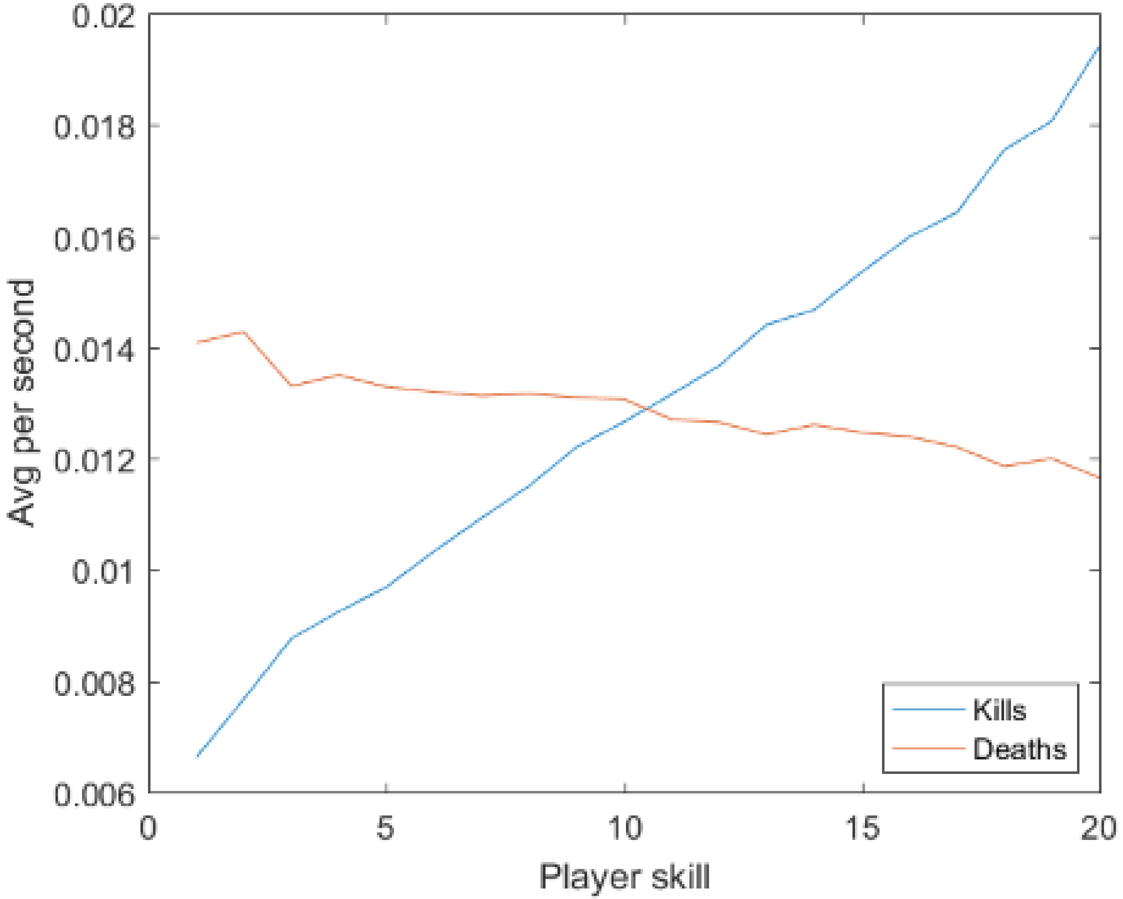
Kills 3: Verify

- Use the same approach for:
  - RTS: Resource spend per minute
  - MOBA: Gold / XP earned per minute
  - CCG: Average Board Mana Advantage
  - Any countable stat after a match

Pre-match Kills per 10 minutes	Predicted win%	Actual win%
0	**52**	**39**
5	**51**	**46**
10	50	51
15	**51**	**57**
20	**53**	**63**

Kills 3: Verify

- Relationship linear
- Linear models will work well



Kills 4: Implementation

Temptation given linear relationship:

1. Fit model to **predict skill rating from kills**
2. Use new prediction to influence skill rating

→ Moving target:

- Changes skills which changes the model
- Devolves to kills defining skill, **changing incentives**.
- Common for games to try this and then back off.
- Inaccuracy makes it worse and worse

TrueSkill® Through Time (TTT)

- The Coalition dissatisfied with common solutions
- Approached Microsoft Research Cambridge
- 2+ year collaboration to significantly improve TrueSkill
- Running in Gears of War 4 since launch
- 343 industries integrating into Halo 5

TrueSkill® Through Time (TTT)

- Microsoft Research: Tom Minka, Yordan Zaykov, et. al
- The Coalition: Ryan Cleven
- Fits skills and (hyper)parameters over all matches jointly
- High accuracy on **already MM** data: 70% vs. ~50%

Game Modes with TTT

- Tracks a skill per game mode, class, platforms, etc.

- Shares skill information between game modes
    - Knows your skill in a new mode before playing that mode
    - No Moving Target Problem: part of the same system

Party Skill TTT

- A skill offset per party size. Few matches required to learn.

- Part of the same model as player skill: not external
  - Partying up? OK, harder matches, but solo skill still estimated right
  - Solo? Ok, easier matches, solo skill estimated right.
  - Skill update accounts for opponents being in parties as well

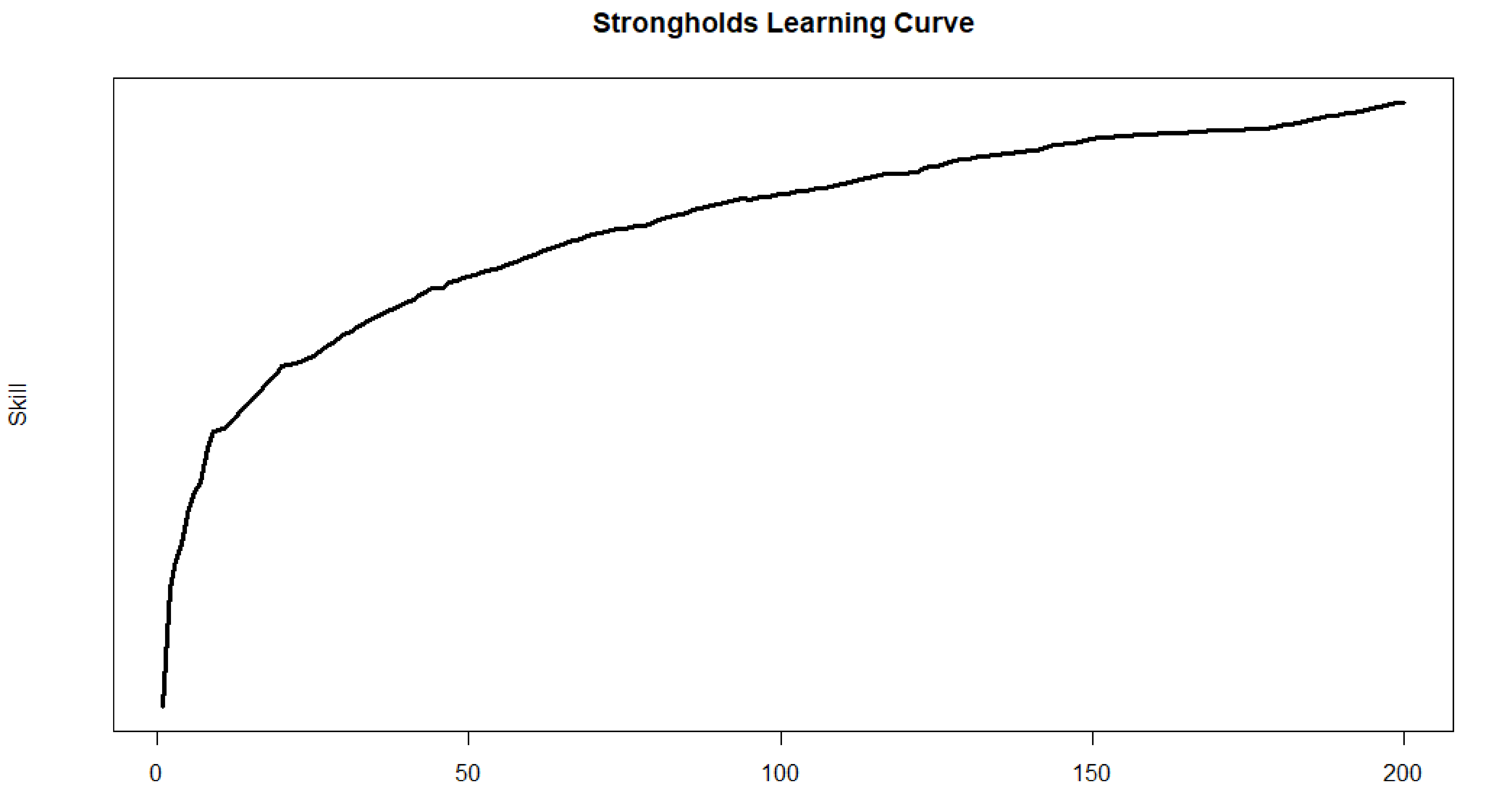
- Learned per **game mode:** organization doesn’t always matter

Party Size Example

Party Size	Original Prediction %	win%	TTT prediction%
1	49	49	48
2	50	50	51
3	49	48	48
4	**53**	**58**	**60**

New Players: TTT

- Learns **best** initial rating, using other modes, classes, etc.
- Finds how fast players catch-up: Learning Curve per Mode
- Learned **simultaneously** WITH skill: no external model
- New player experience is fair, should result in **less churn**


Strongholds Learning Curve
Skill
0
50
100
150
200

New Players with TTT

Games Played	old prediction%	win%	TTT prediction
first game	**49**	**44**	**44**
1	**49**	**45**	**45**
2	**49**	**46**	**46**
3	**49**	**47**	**46**
4	**49**	**47**	**47**
5	**49**	**47**	**47**
6	49	48	47
7	49	48	48
8	50	48	48
≥9	50	51	51

New Players In Gears of War 4

Games Played	Win% Before	Win% After
First Game	40	50
1	42	50
2	43	49
3	43	50
4	44	49
5	45	49
6	45	49
7	45	50
8	46	49
≥9	48	49

Kills and Other Counts with TTT

- Don’t have a match’s kills *before* a match

- Instead, put kills on the *output* as something we predict
  - Knowing what happened after improves skill estimate

- Update a single skill rating based on predicting both:
  - Win %
  - Kills per minute

Kills with TrueSkill Through Time

- Still enforces that the winning team overall did better (incentives)
- Losing players can outperform winners
- Still just ONE skill rating per player
- Halo 5: |avg(kills) – avg(predicted)| **< 0.02**

Kills with TrueSkill Through Time

Pre-Match Kills per 10 minutes	Predicted win%	Actual win%	TTT Prediction%
0	**52**	**39**	**39**
5	**51**	**46**	**45**
10	50	51	53
15	**51**	**57**	**58**
20	**53**	**63**	**62**

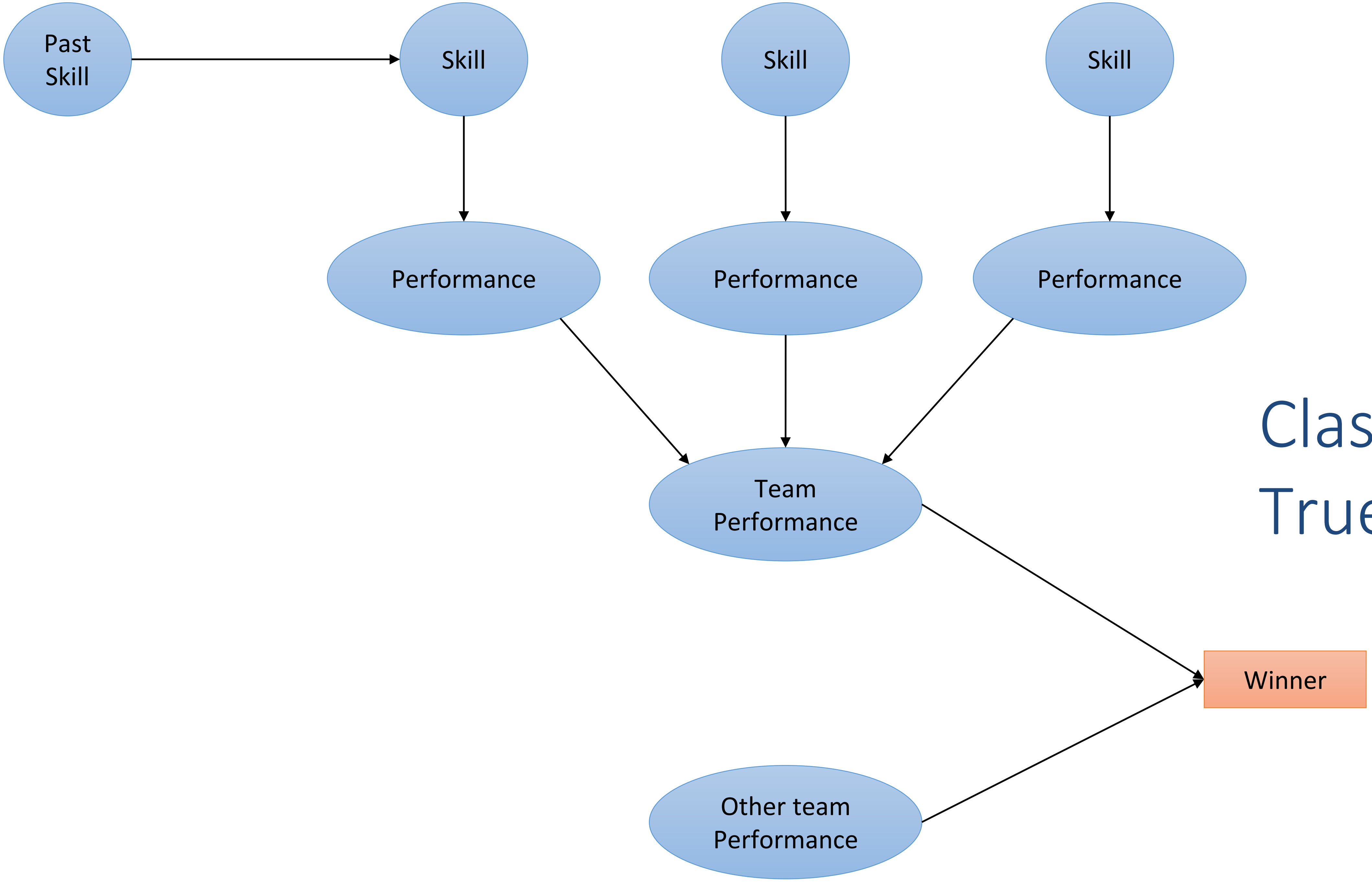
Use for Any Event Count

- Event count examples to verify:
  - **RTS**: Resource spend per minute
  - **MOBA**: XP or Gold per minute
  - **CCG**: Average board mana advantage
  - **Soccer**: Field Coverage per game, avg. distance from goals

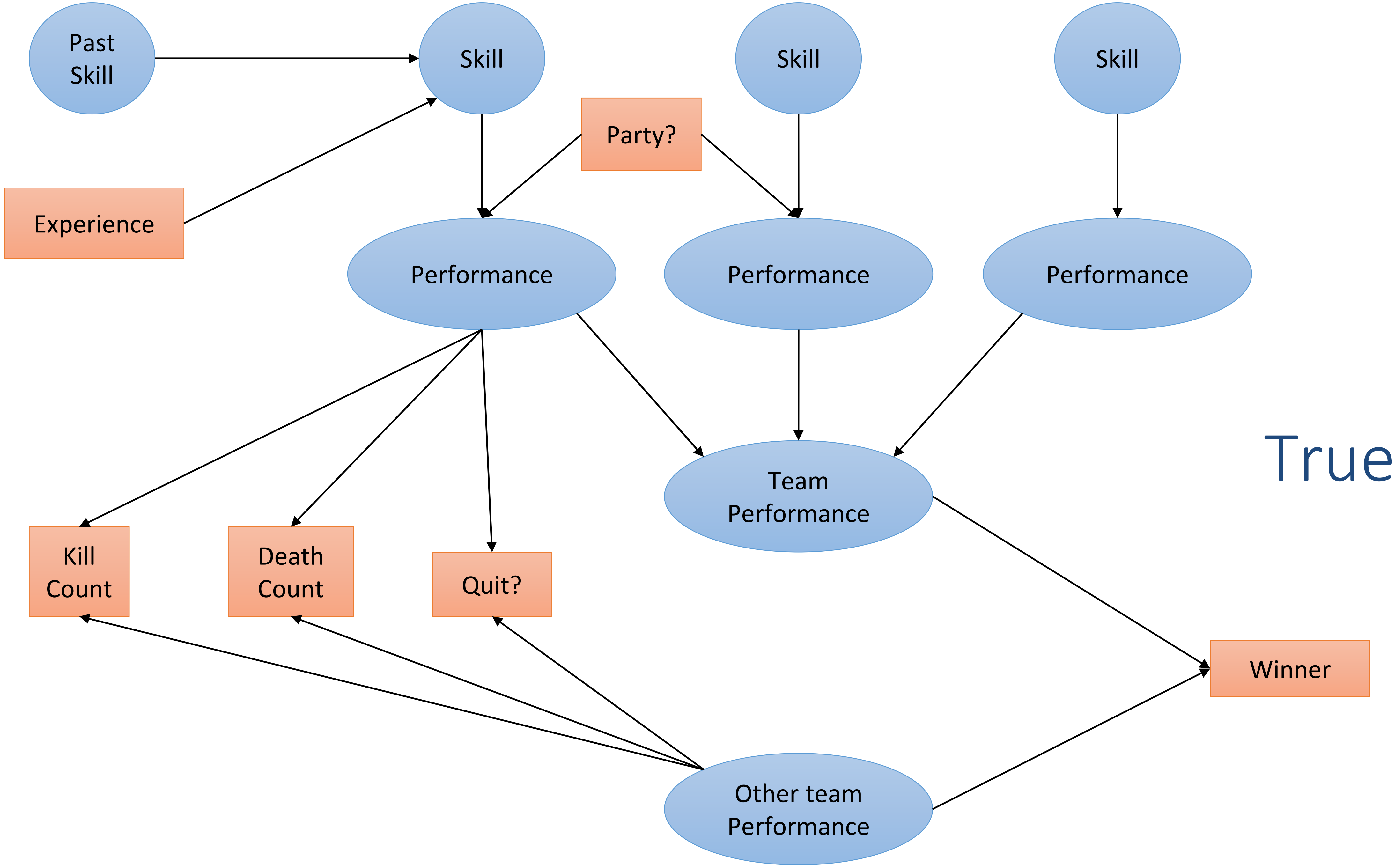
- Per **Class:**
  - Verify correlated with existing skill ratings
  - E.g. prefix event names with the hero: (Rogue_Kills, Tank_Kills, …)

Other TTT Benefits

- Smurf Detection:
  - Throws anomalies if players kill, die, heal, resource too much
- Handles **bot** skill correctly
  - Use them to accurately find new player skills
  - GoW 4 uses skill with their bots

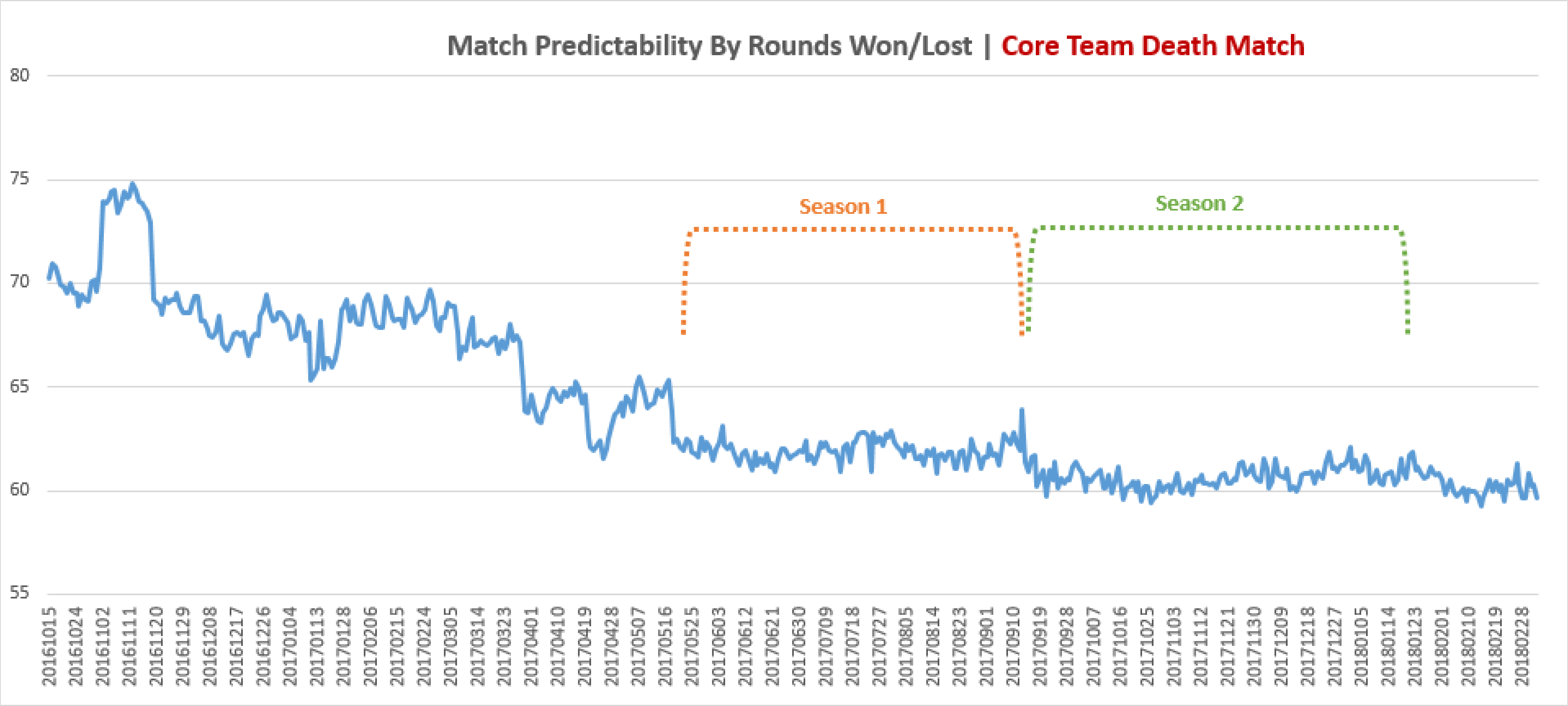


Classic TrueSkill


Past Skill
Skill
Skill
Skill
Experience
Party?
Performance
Performance
Performance
Team Performance
Kill Count
Death Count
Quit?
Winner
Other team Performance

TrueSkill2

Gears 4 Improvement Over Time



Apply Steps to Your Game

1. Brainstorm with your Developers
   - Designers, Engineers, Producers, anyone might have a good idea
   - Came up with 5 metrics in 5 minutes

2. Slice on those features and metrics
   - Just like we did in the examples
   - Check for cases where predicted win% is different than actual win%

3. Integrate ones that you should
   - Ideally using something like TrueSkill Through Time
   - Learn everything simultaneously

TrueSkill Through Time in the Cloud

- TTT uses data from ALL our matches from the beginning
- Runs in parallel in the cloud on many machines
- Heavily optimized by Microsoft Research
- Should we add as a service from the cloud gaming team?

Questions? Also References.

- Elo: wikipedia.com/wiki/Elo_rating_system
- Glicko: glicko.net/glicko.html
- Trueskill: https://www.microsoft.com/en-us/research/publication/trueskilltm-a-bayesian-skill-rating-system/
- TrueSkill2: https://www.microsoft.com/en-us/research/publication/trueskill-2-improved-bayesian-skill-rating-system/
- Contact for links (I'll also tweet them out):
    - twitter.com/joshua_menke, reddit: ZaedynFel
- Further Discussion: Overlook 3022