PURDUE UNIVERSITY

Understanding Game Balance in Mosaic Warfare with Explainable Artificial Intelligence

SERC AI4SE & SE4AI Workshop

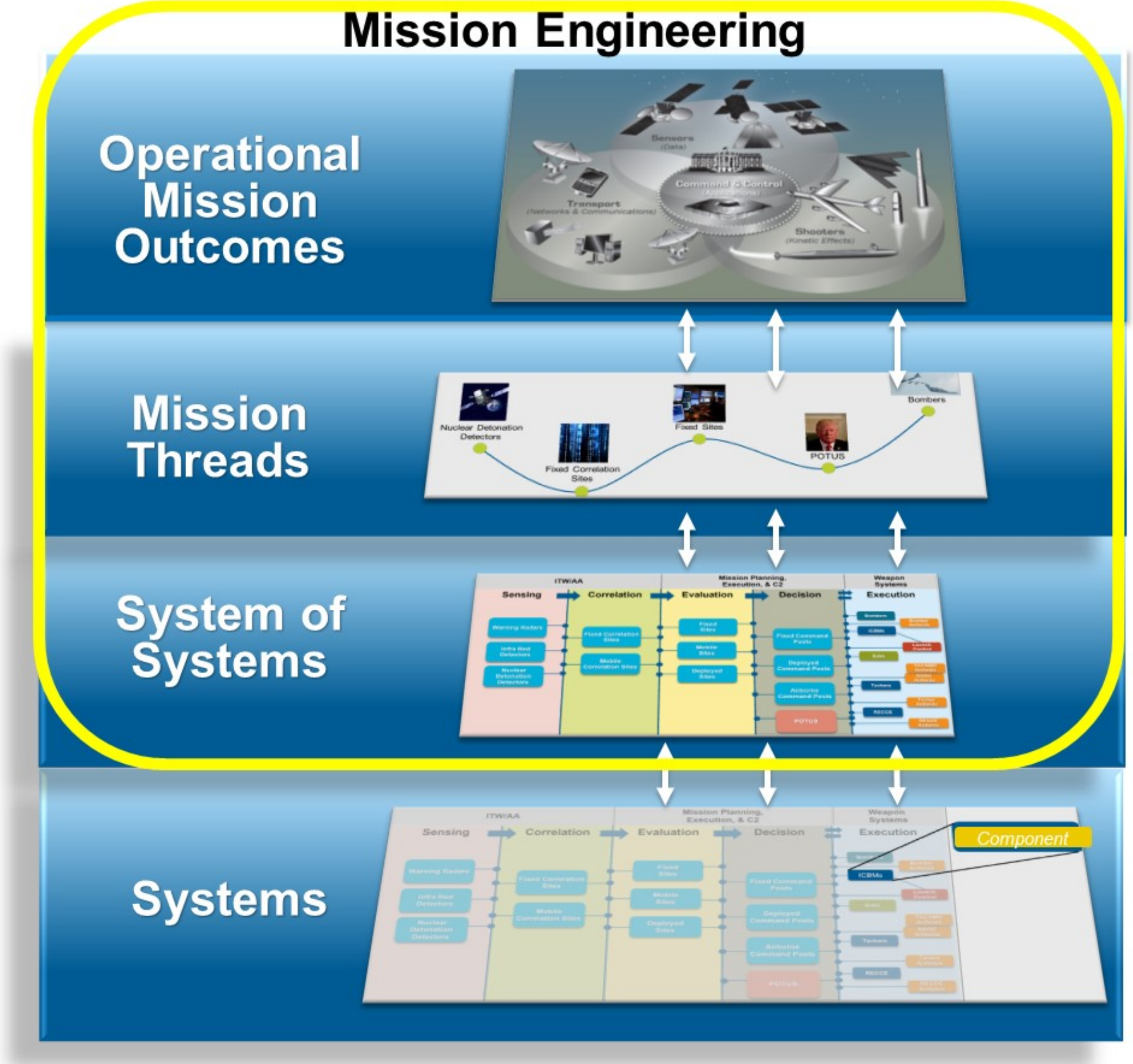
Dr. Daniel DeLaurentis

Co-Authors:

Drs. Ali Raz, Jitesh Panchal, Adam Dachowicz and Kshitij Mall

October 28, 2020

Mission Engineering: Top Level View of Relationships



In this view, systems engineering applies at multiple levels

- Systems and components
- Systems of systems
- Operational missions

Mission engineering is considered ‘above the system level’, addressing systems of systems in a mission context

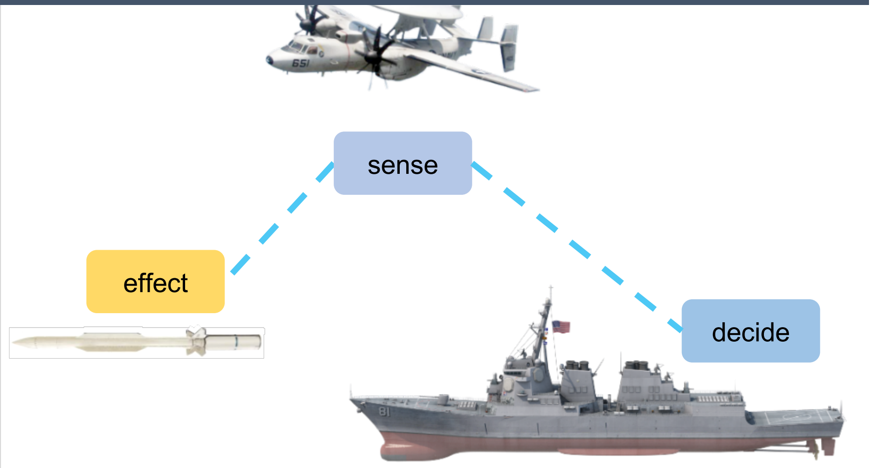
‘Mission Threads’ provide linkage between SoS/ systems and operational mission

The Pathway to Mosaic Warfare

Courtesy DARPA STO

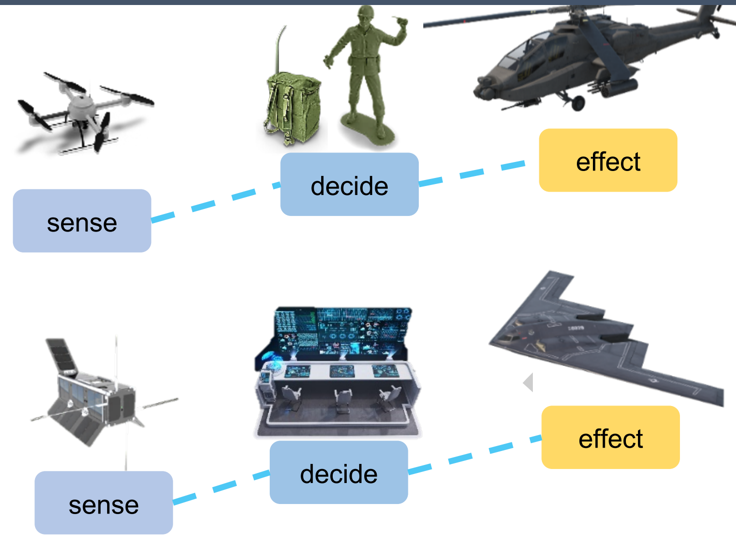
Distributed Kill Chain

Manual integration of existing systems



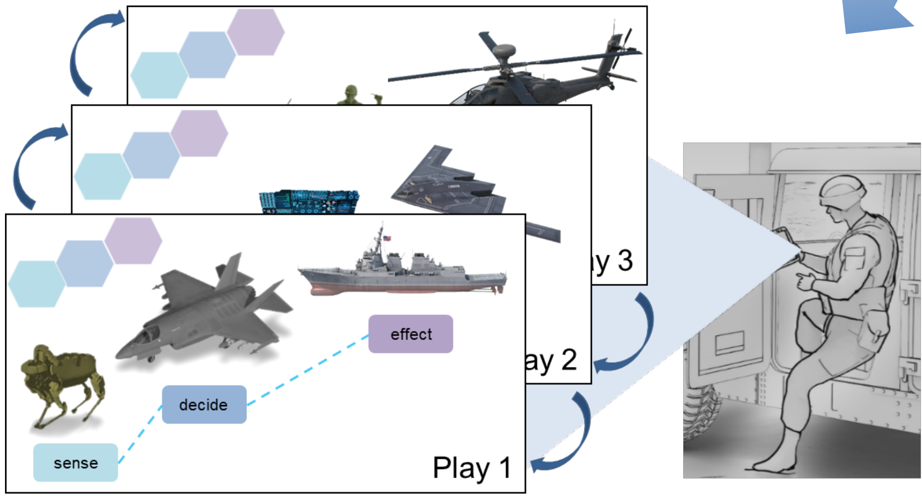
System of Systems

Automated design and integration to accelerate SoS fielding



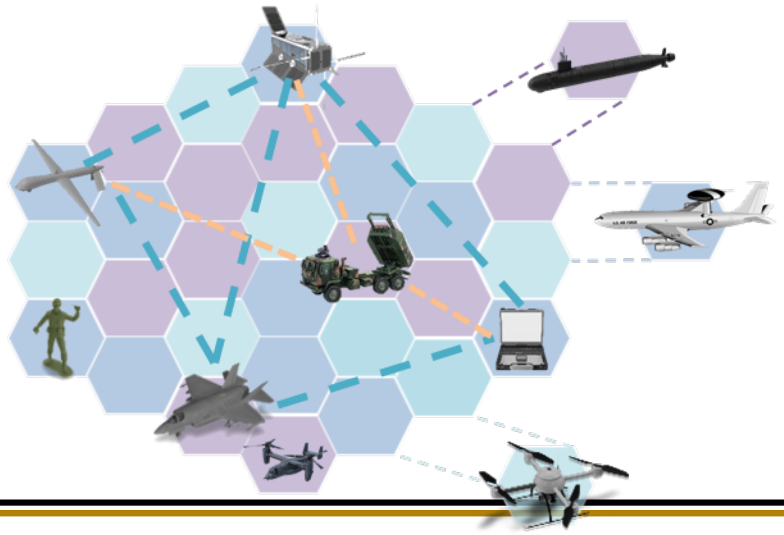
Adaptive Kill Webs

Automation to operate SoS at scale and adapt missions



Mosaic Warfare

Ability to compose new effects webs at campaign time with any available weapon system

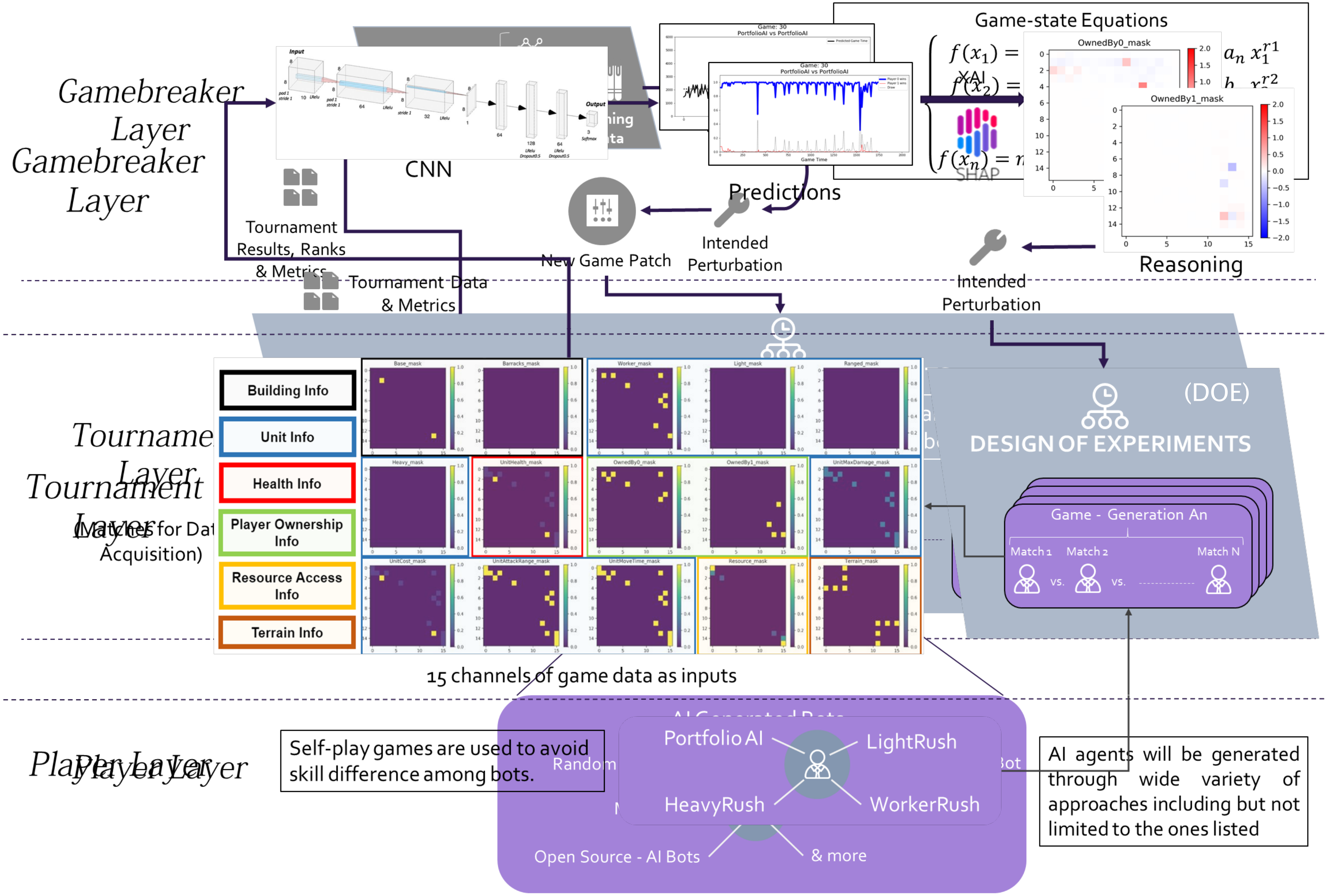


Towards Mission Engineering for Mosaic Warfare via Real-time Strategy Games- A Research Challenge

- How do we transition from capable systems to intelligent and adaptive system of systems architectures?
  - How capabilities of systems along with initial conditions and their interactions impact the battle outcome?
  - How does adversary's systems along with initial conditions and their interactions impact the battle outcome?
  - Is there a balanced playing field? If yes, what architecture and sequence of actions will make it unbalanced? In who's favor?

- Understanding how to win in Mosaic warfare via Real Time Strategy (RTS) Games + AI
  - DARPA's Gamebreaker Program
  - How do new capabilities, rules, and modifications affect game balance?
  - How can a game balance equation be developed?
    - ***Purdue's approach:*** *Learning to Gamebreak (L2G) via AI/ML and XAI*

Learning 2 Gamebreak (L2G) Framework:



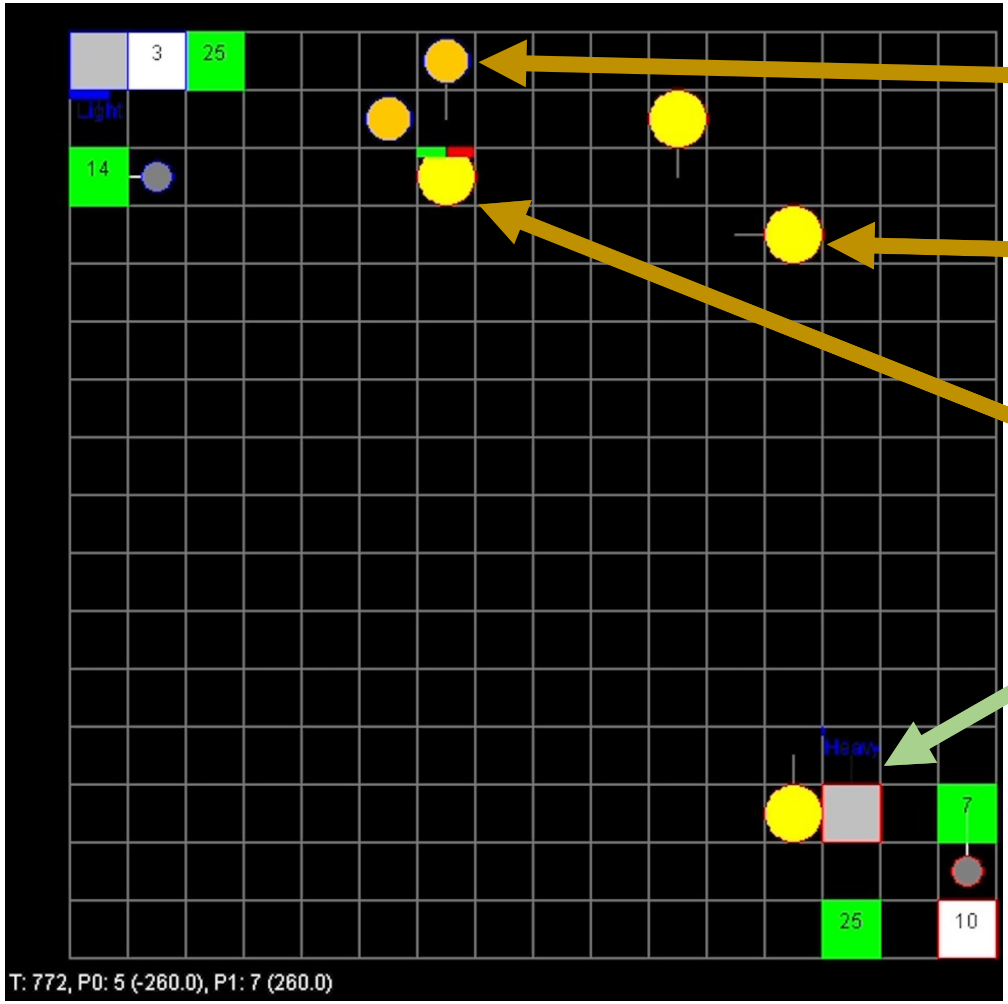
- **Goals**:
  1. From tournament games, identify game balance metrics and game features that impact game balance.
  2. Refine DOE to acquire new information.
- **Tools**:
  - ➢ CNN + MCDN, SHAP

- **Goals**:
  1. Run games between players according to specified DOE.
  2. Pass actionable data to Gamebreaker Layer.
- **Tools**:
  - ➢ MicroRTS, in-house code

- **Goal**:
  - Specify game bots that will compete in Tournament Layer.
- **Tools**:
  - ➢ Existing player "bots" implemented in MicroRTS

MicroRTS Gameplay

Game Map



Light Unit
(Fast, Low Damage)

Heavy Unit
(Slow, High Damage)

Damaged Heavy Unit

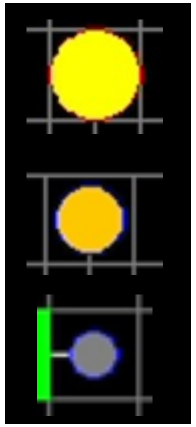
Barracks (Builds military units with resources)

Resources

Worker (Gathers resources)

Base (Builds workers with resources)

Units



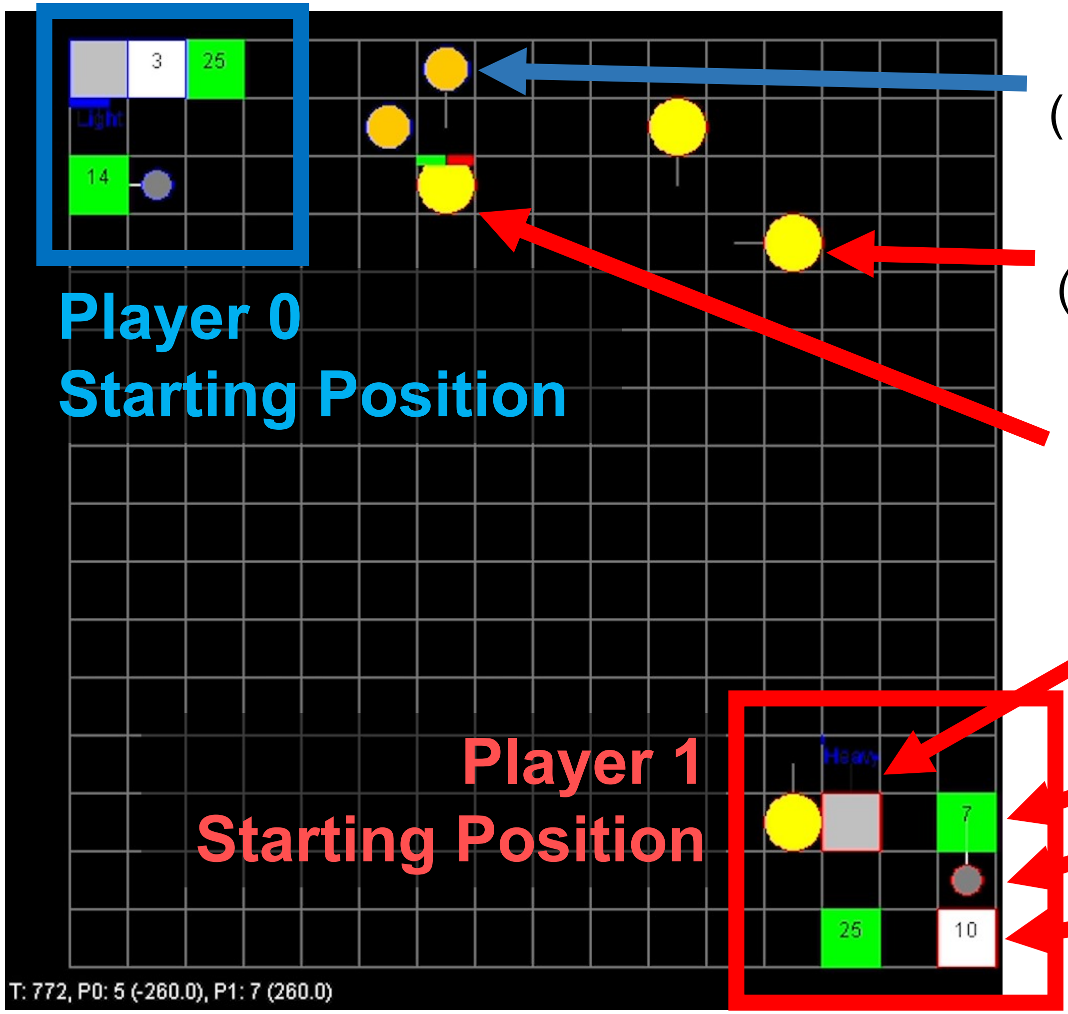
Heavy

Light

Worker

MicroRTS Gameplay (Cont…)

Game Map



Light Unit (Fast, Low Damage)

Heavy Unit (Slow, High Damage)

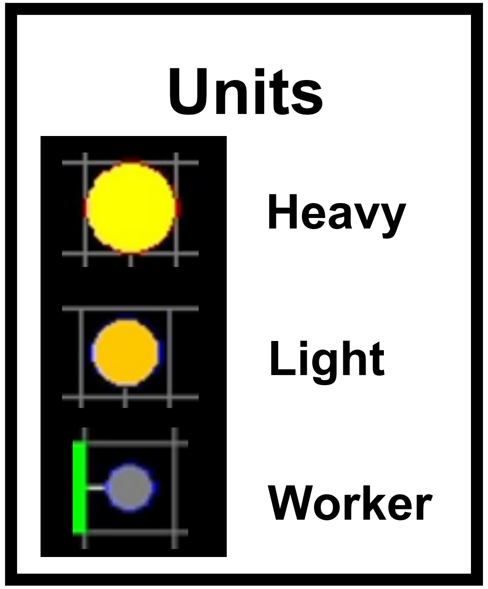
Damaged Heavy Unit

Barracks (Builds military units with resources)

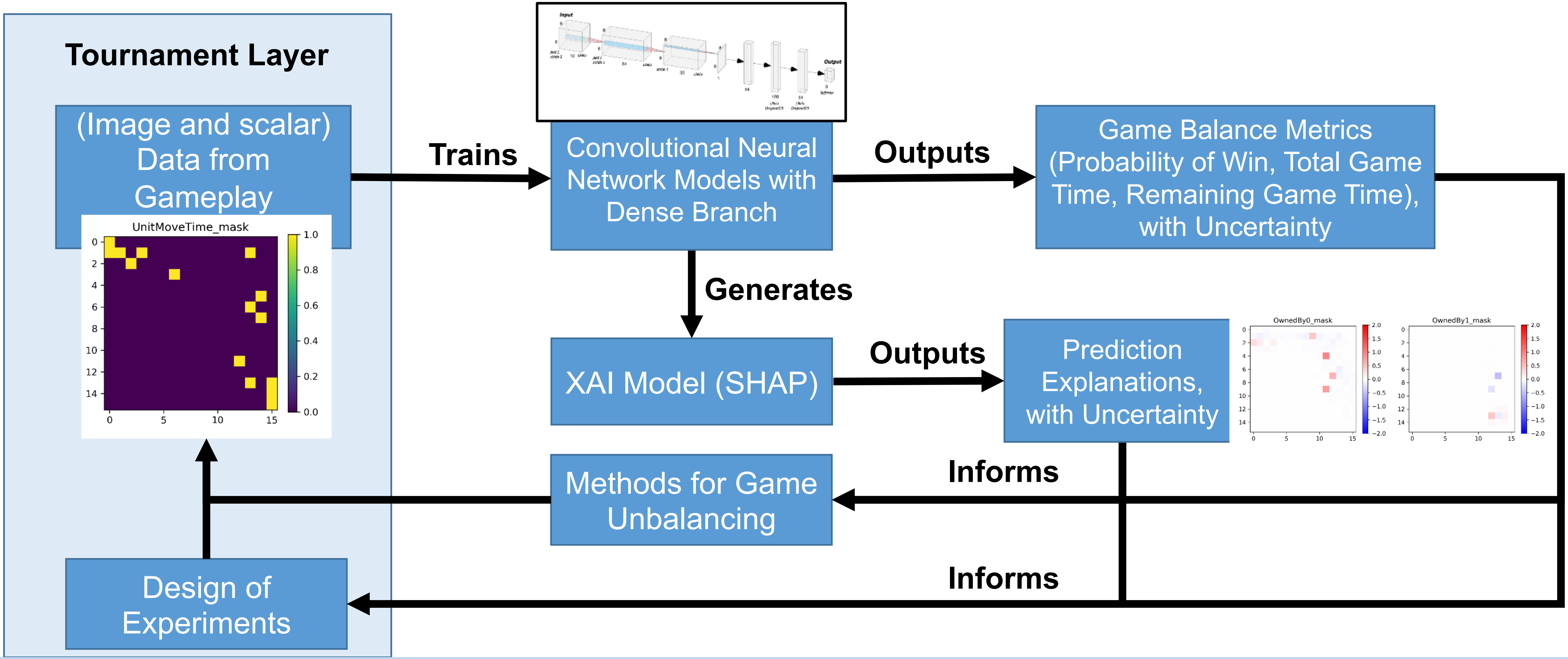
Resources

Worker (Gathers resources)

Base (Builds workers with resources)



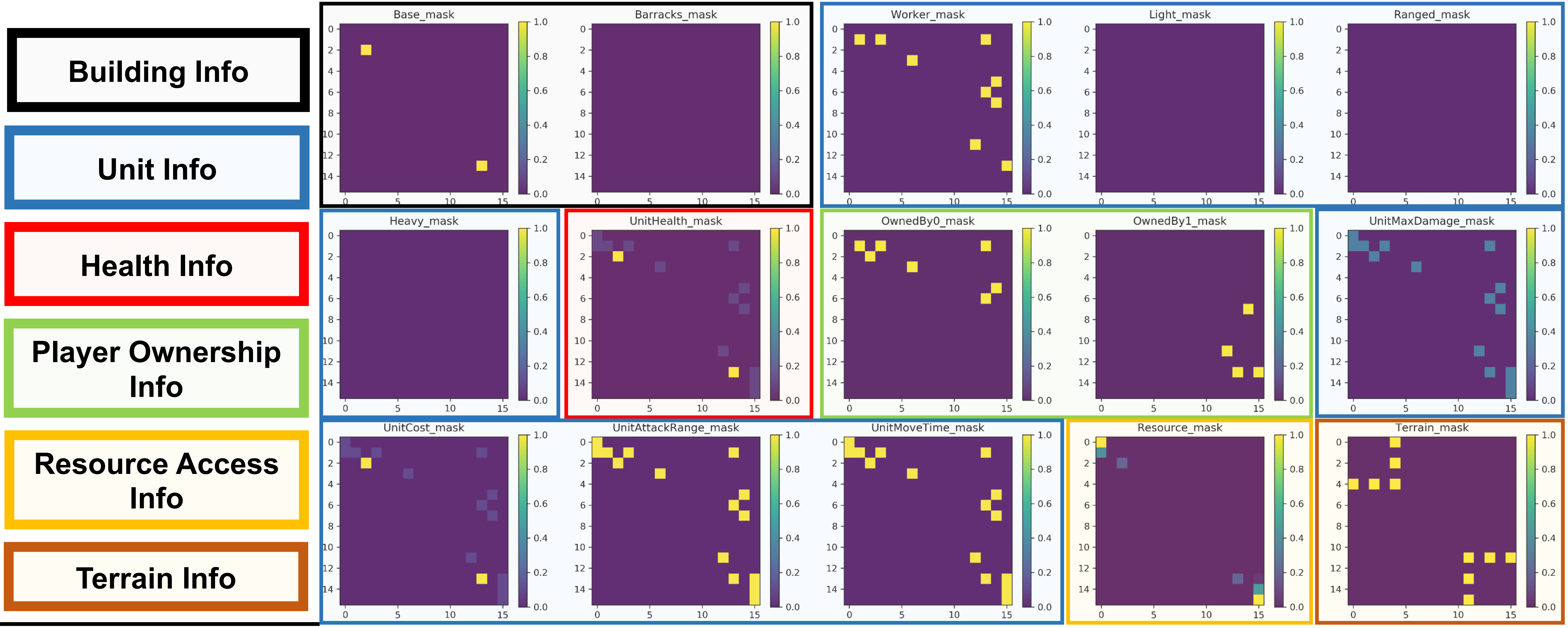
L2G Framework Implementation: Approach



Each “block” can be extended as we increase the game complexity.

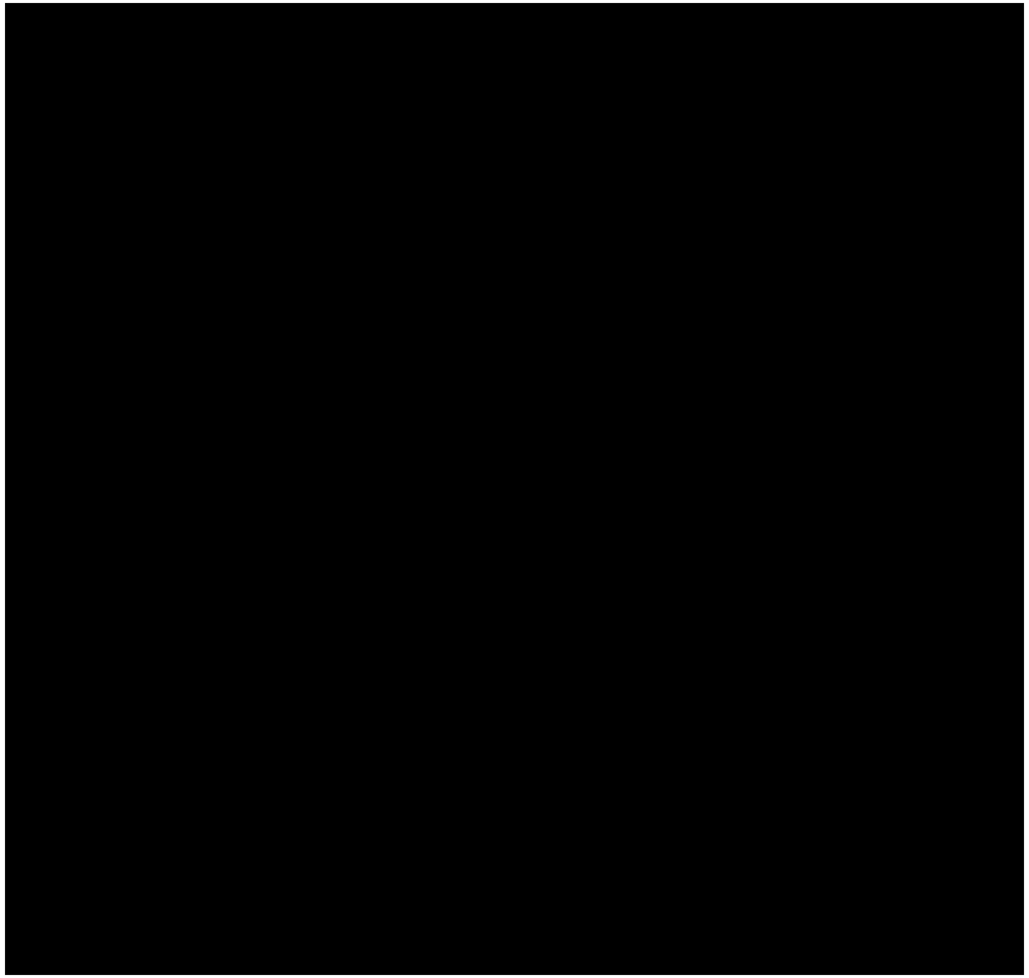
Input Data: Geographic Features

From each game, at each time step, generate 15 image “channels” encoding spatial information about the game and players

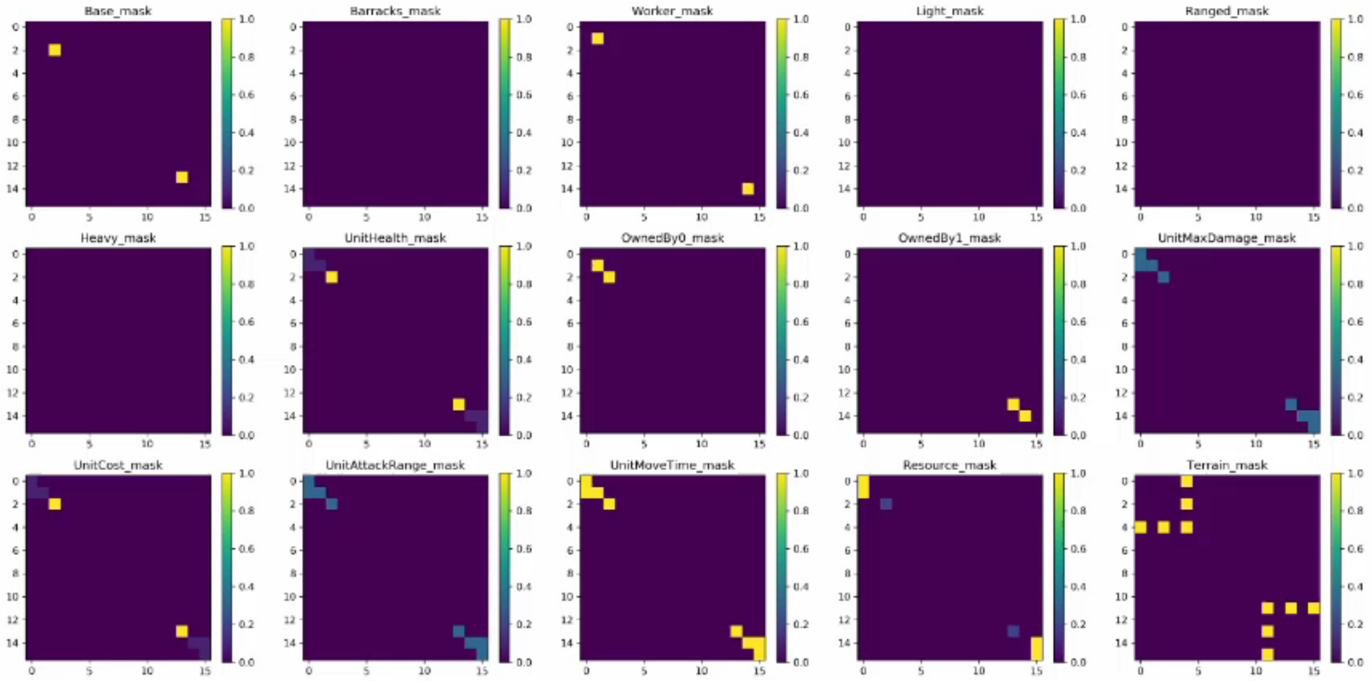


Input Data: Image Features and Example Gameplay

PortfolioAI vs PortfolioAI

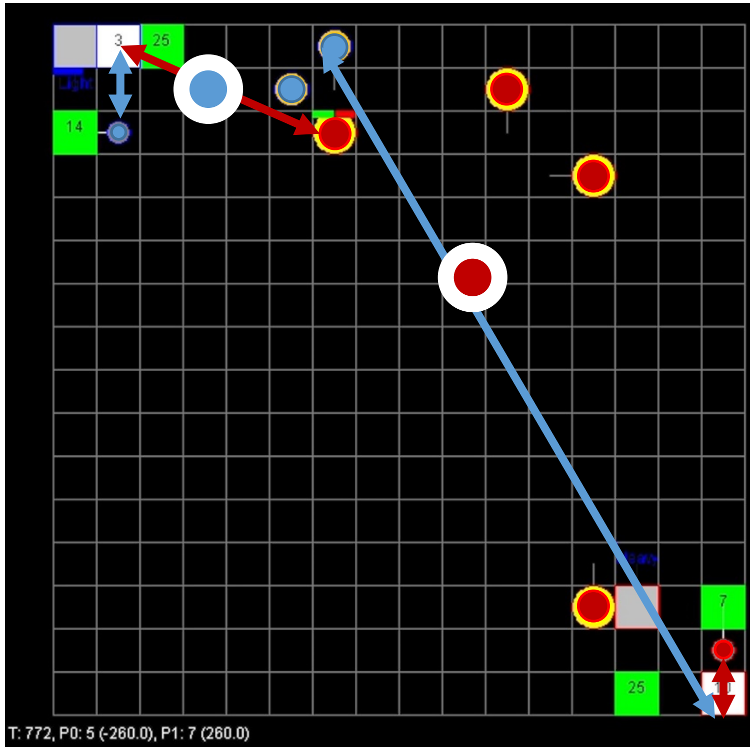


Extracted data for current game frame



Input Data: Non-Geographic Features

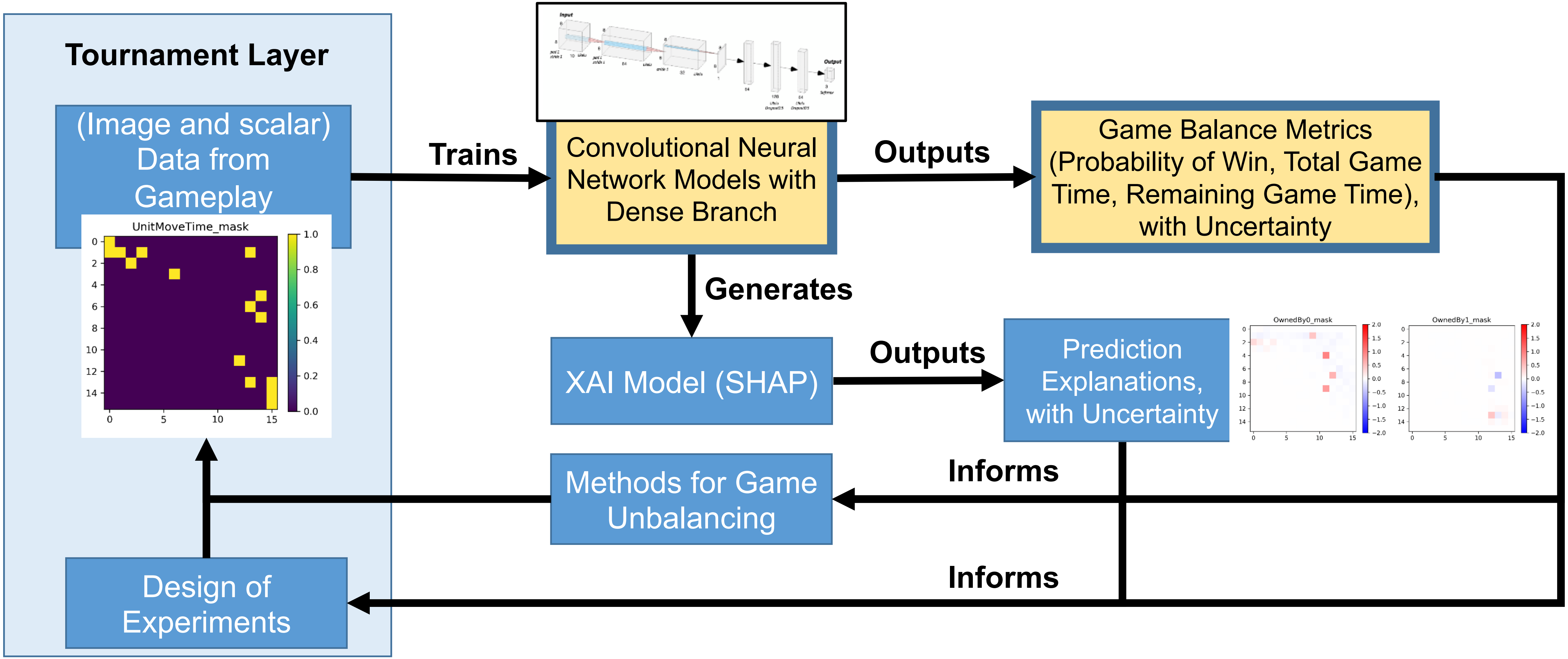
Type of Feature for Two Players	Number of Features
Minimum distance between units and bases	4
Location of unit *centroids**	4
Total health	2
Total units	2
*Damage Rate***	2
Terrain	1
Grand Total	**15**



*Two coordinates for each player

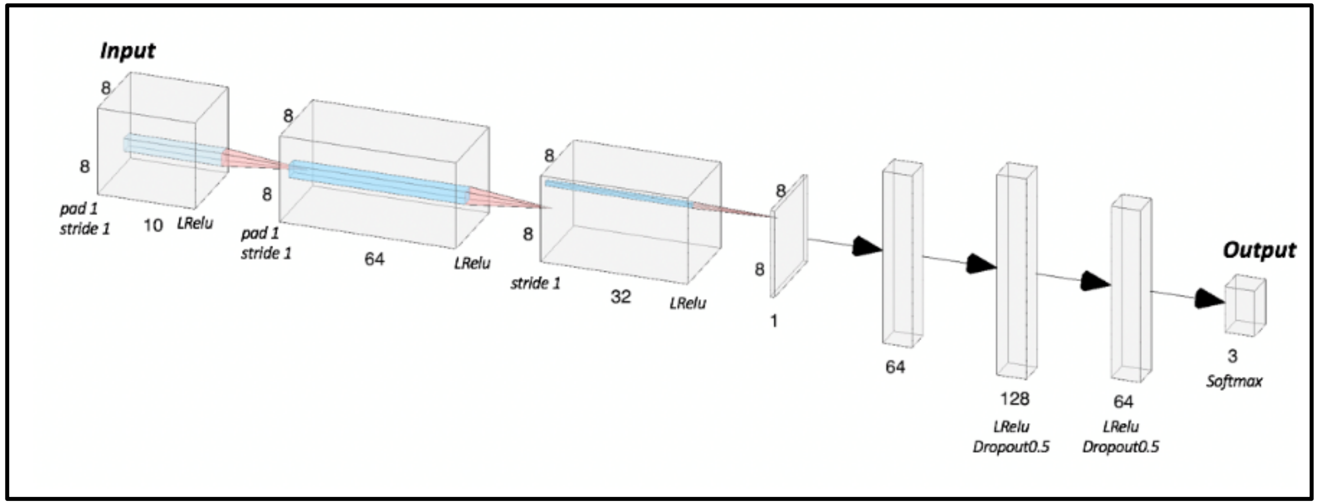
**Total damage dealt to the player divided by current game time

L2G Framework Implementation: CNN Model and Uncertainty



Convolutional Neural Networks (CNNs)

- Machine learning technique useful for analyzing data with spatial structure
  - Convolutional layers extract features that preserve spatial information
- Previous literature uses CNN-based approaches to analyze MicroRTS gameplay [1]
- **Pros:** applied to image input data
- **Cons:** outputs are difficult to explain

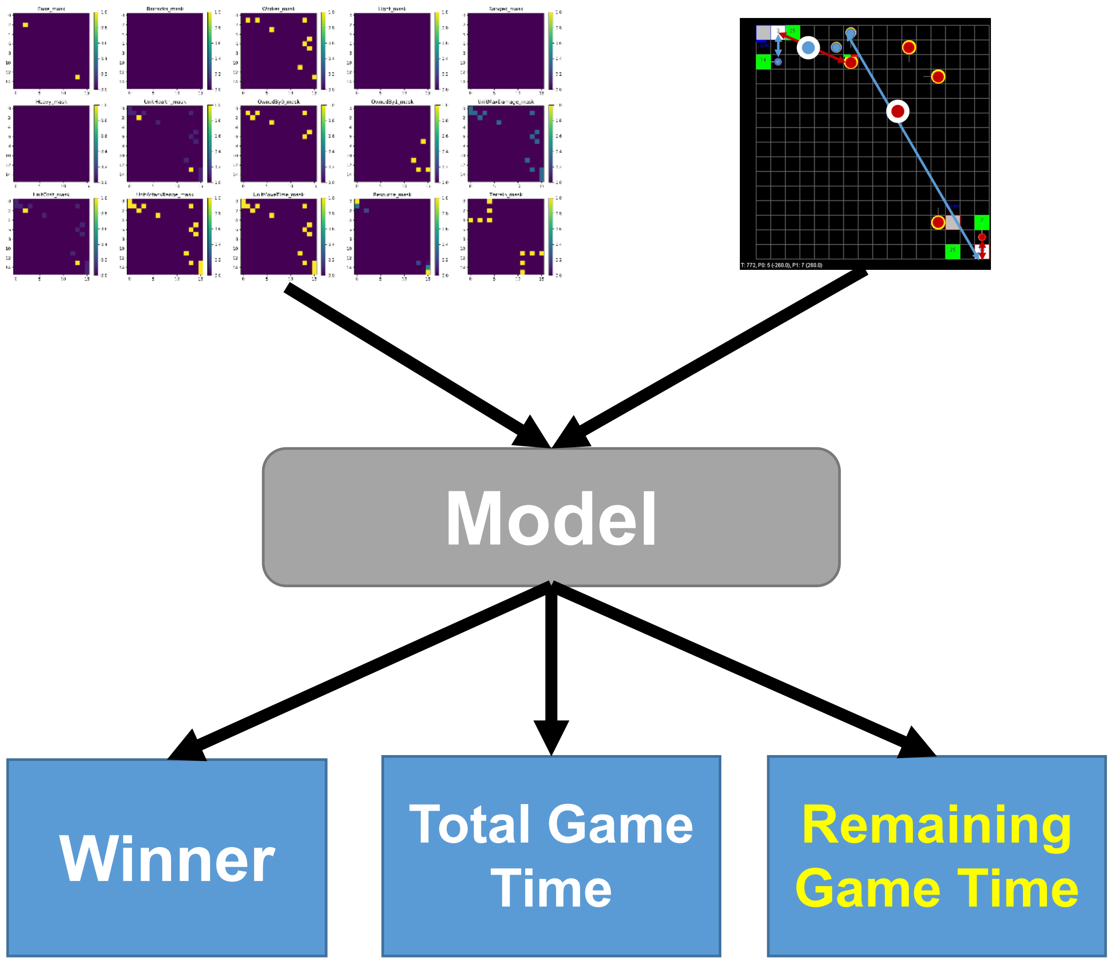


[1] Stanescu, Marius, et al. "Evaluating real-time strategy game states using convolutional neural networks." *2016 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 2016.

CNN Model Overview

- **Model Inputs***
  - ➢ Spatial data (passed through convolutional layers)
  - ➢ **New:** Non-geographic data (passed through dense layers)
- **Model Outputs**
  - ➢ Probability of each player winning
  - ➢ Predicted total game time
  - ➢ **New:** Predicted remaining game time

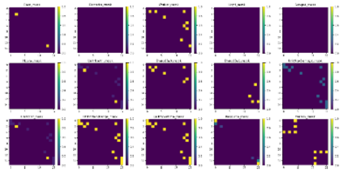
*Taken from each game time step

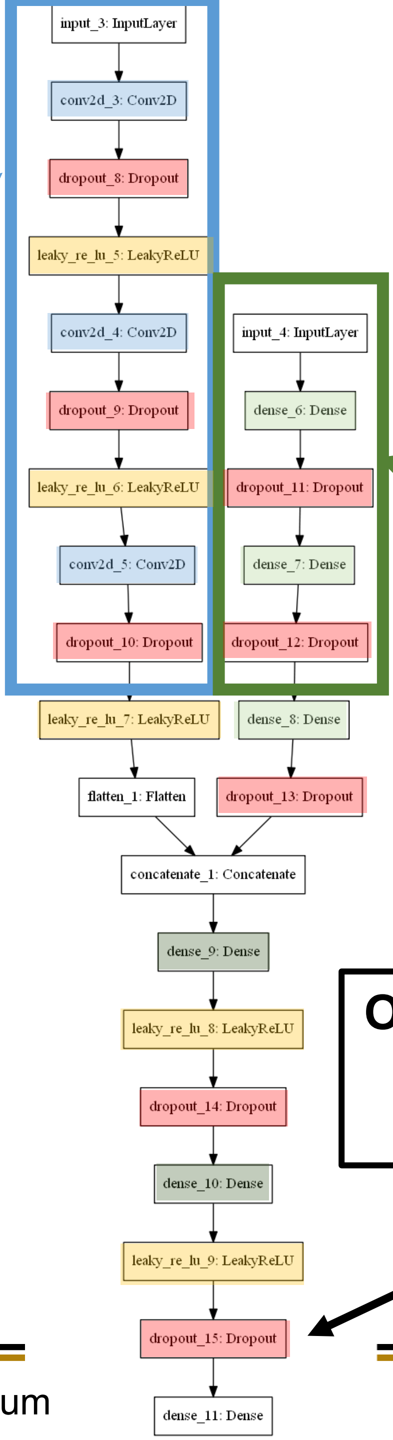


CNN Model Architecture

- **DOE**: Training data from self-play games between Portfolio AI players
- **Inputs**: both geographic and non-geographic data
- **Uncertainty Quantification**: Implements MC-Dropout for all convolutional and dense layers
- **XAI**: MC-Dropout samples are also used to train SHAP explanation models

Input: Image data





Convolutional layer

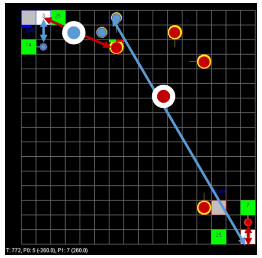
Leaky ReLU Activation

Dense Layer + ReLU Activation

Dense Layer + Linear Activation

Dropout Layer

Input: Non-spatial data

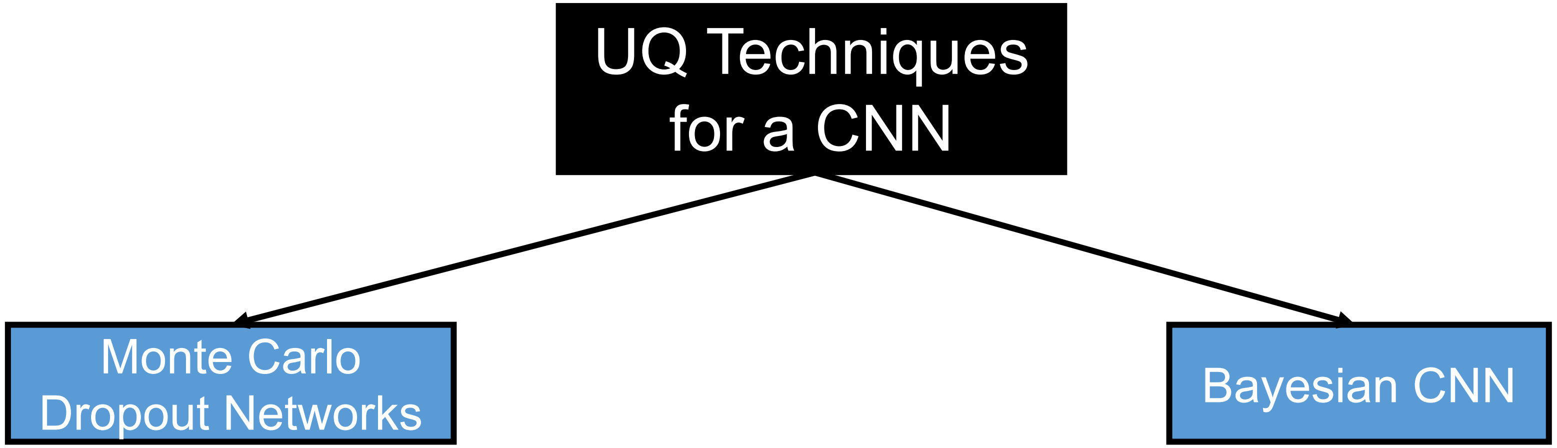


Output: Winner predictions OR Time predictions

[1] Gal, Yarin, and Zoubin Ghahramani. "Dropout as a bayesian approximation: Representing model uncertainty in deep learning." *international conference on machine learning*. 2016.
[2] Gal, Yarin, and Zoubin Ghahramani. "Bayesian convolutional neural networks with Bernoulli approximate variational inference." *arXiv preprint arXiv:1506.02158* (2015).

Incorporating Uncertainty Quantification in CNN Model

- **Uncertainty Quantification (UQ)**
  - Characterizes robustness of the current CNN model
  - Informs data collection for improving the model performance
- **Monte Carlo Dropout Networks** (MCDNs) [1] currently the **state of the art** for epistemic uncertainty



Monte Carlo Dropout Networks:

- More efficient computation
- Dropouts in both testing and training data
- Dropouts before every weight layer

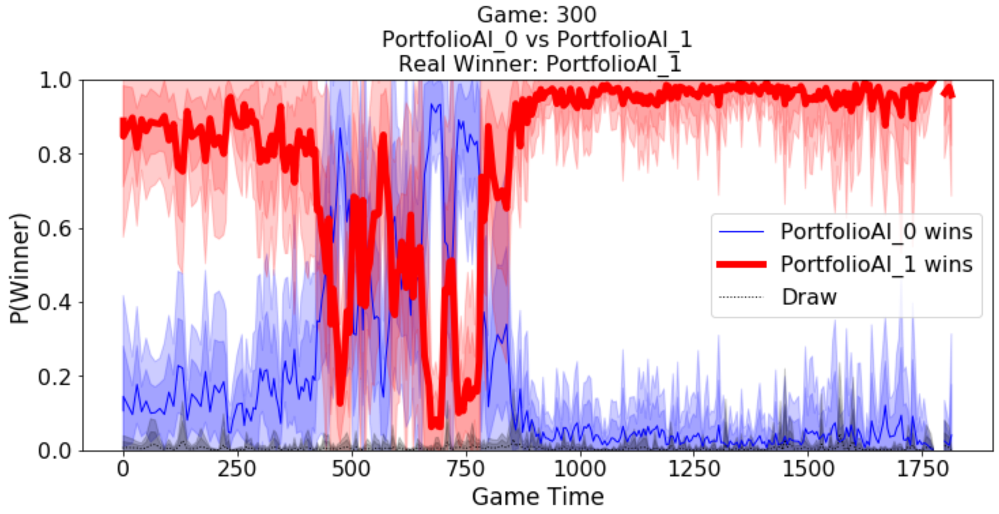
Bayesian CNN:

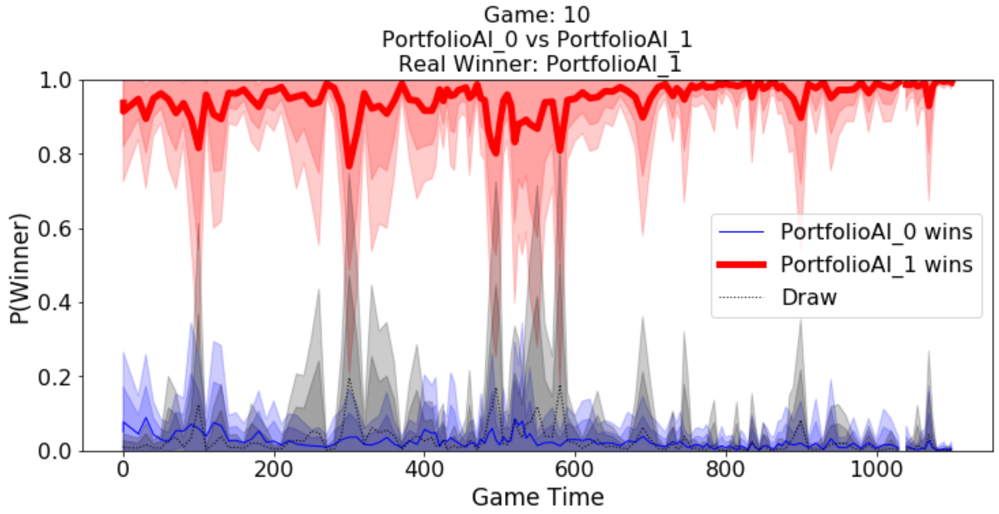
- Computationally prohibitive

[1] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In international conference on machine learning, pages 1050–1059, 2016.

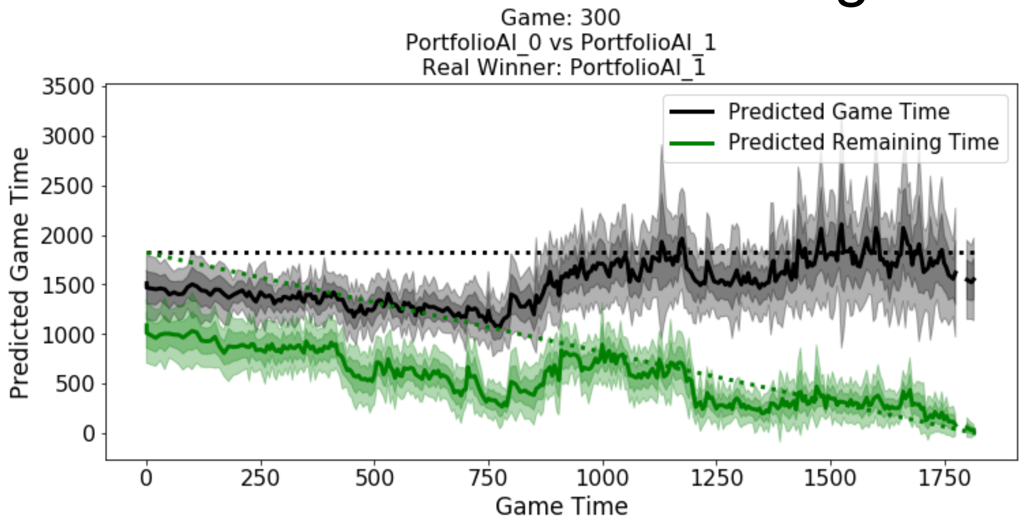
Model Outputs: An Example

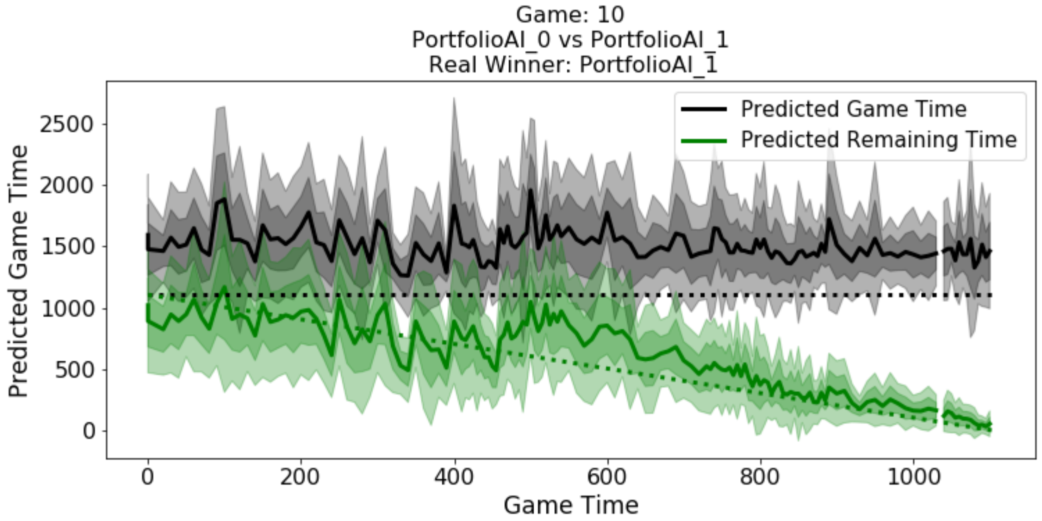
Probability of Win over Time





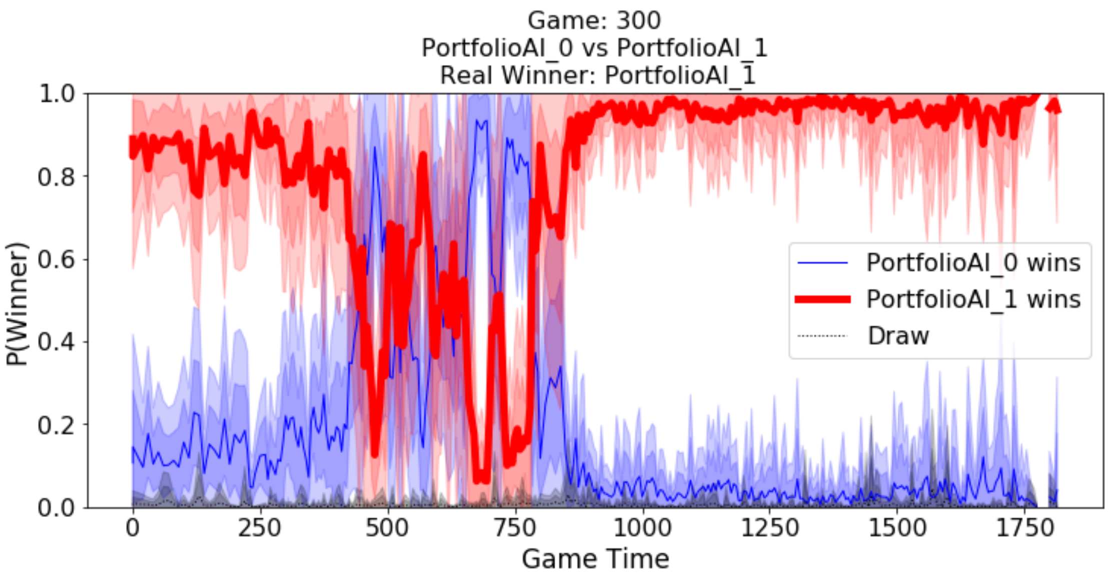
Predicted Total and Remaining Time



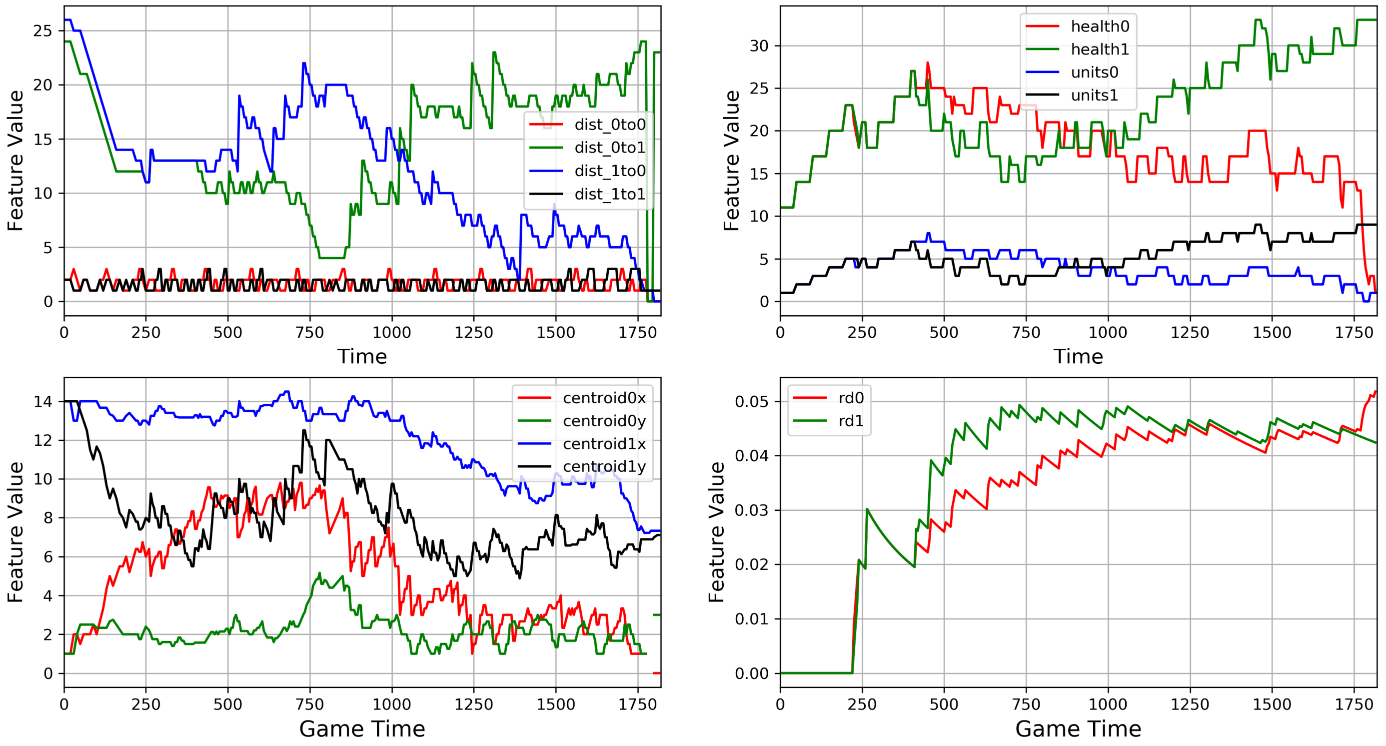


Model Outputs: An Example (Cont…)

Probability of Win over Time

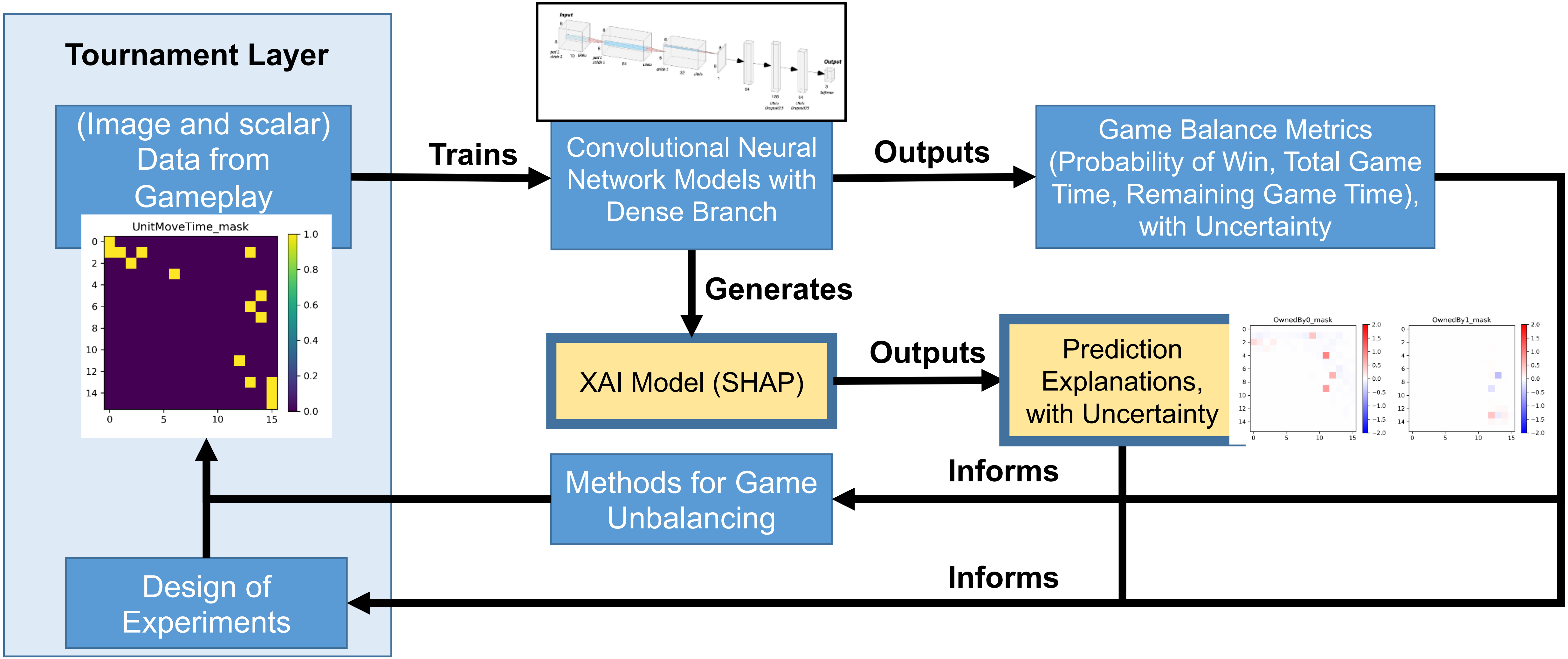


Features over Time



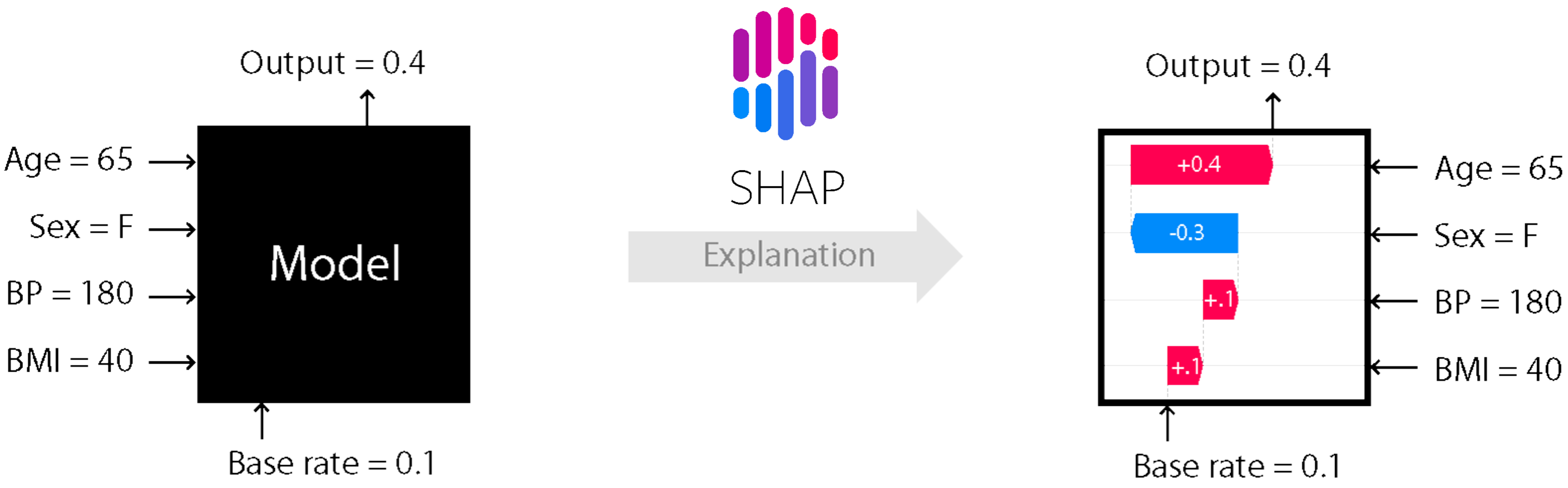
Examining model outputs and feature inputs over time allow users to **identify relationships between balance metric predictions and the current game state**.

L2G Framework Implementation: XAI and Uncertainty



SHapley Additive exPlanations (SHAP)

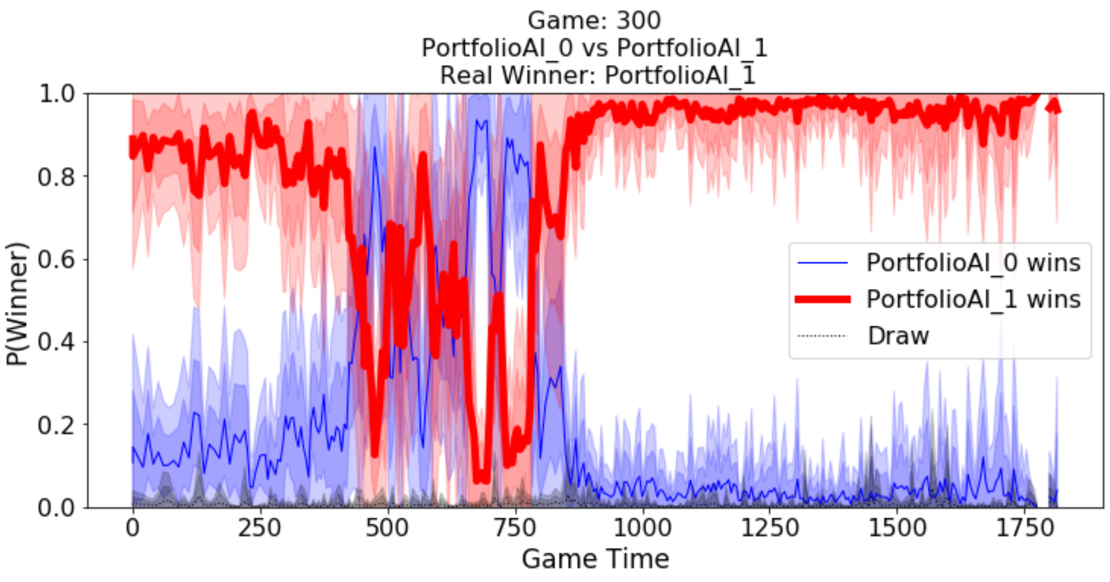
- Based on Shapley values first introduced by Dr. Lloyd Shapley in 1953
- Currently the **state of the art** for reverse engineering the output of any predictive model
- Tells **which features** are **more relevant** for a prediction or for a model as a whole
- Focuses on **coalitions** in **cooperative game theory**: how do **individual features** contribute to the overall **prediction?**



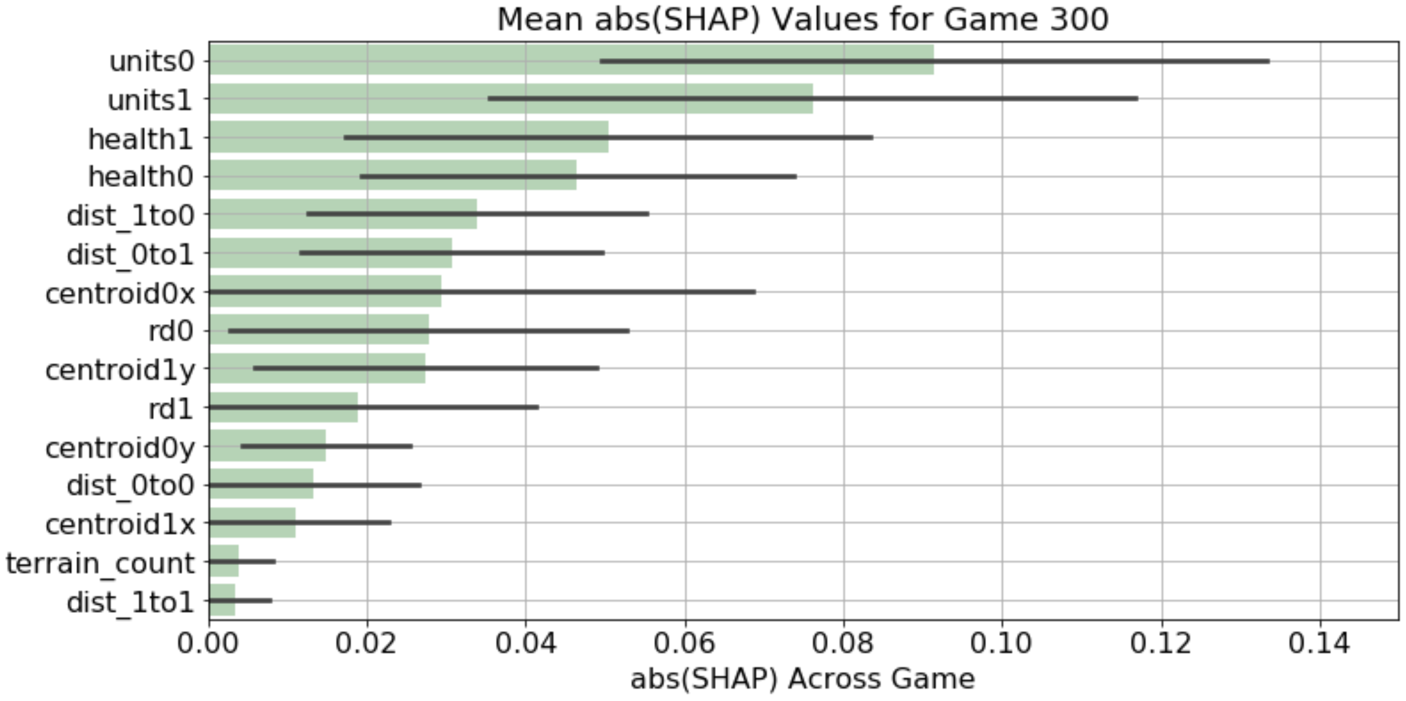
Gamebreaker Dashboard and Analysis Demo

SHAP Outputs: An Example

Probability of Win over Time

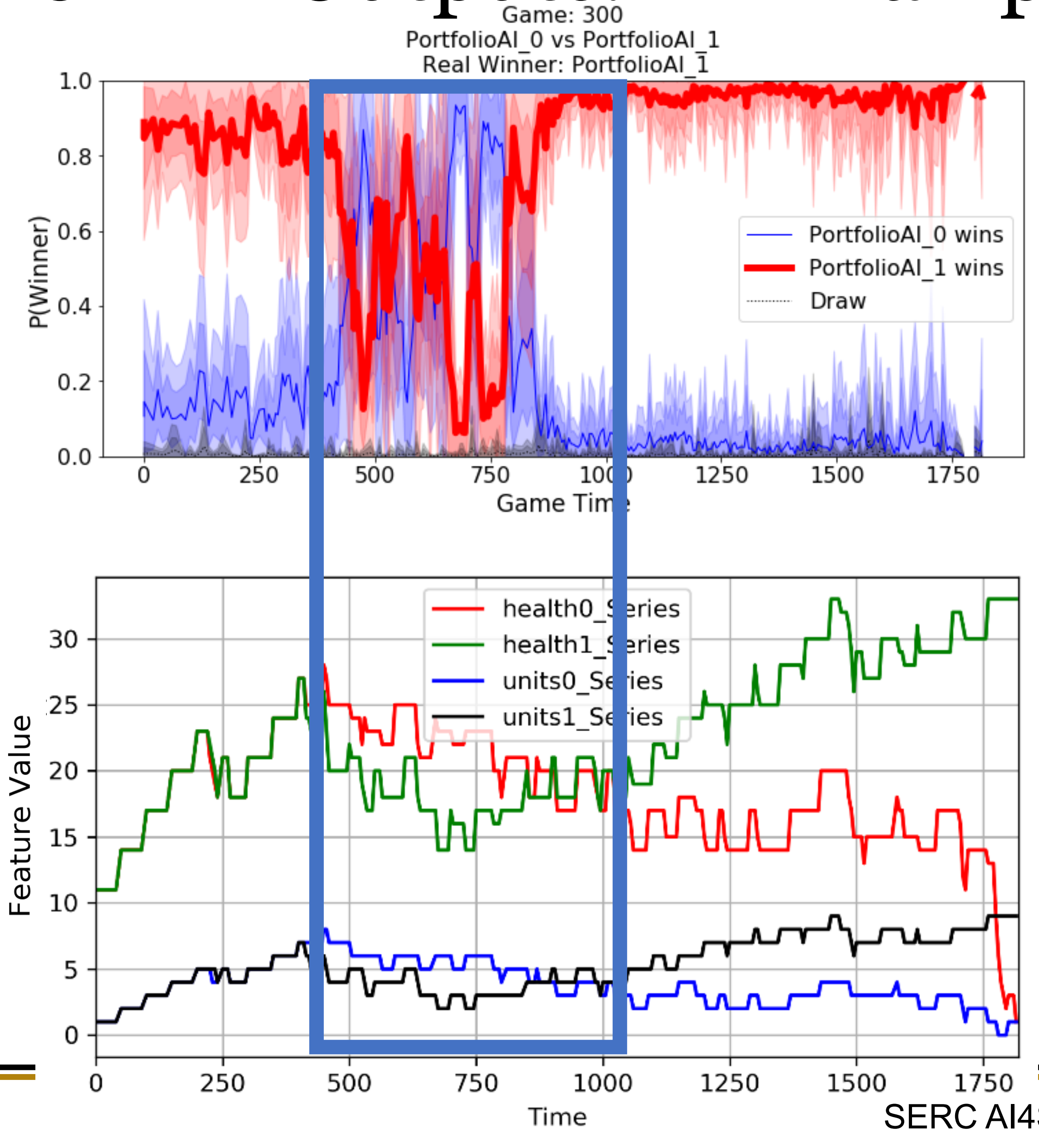


Mean |SHAP|,
For Player 0 Winning

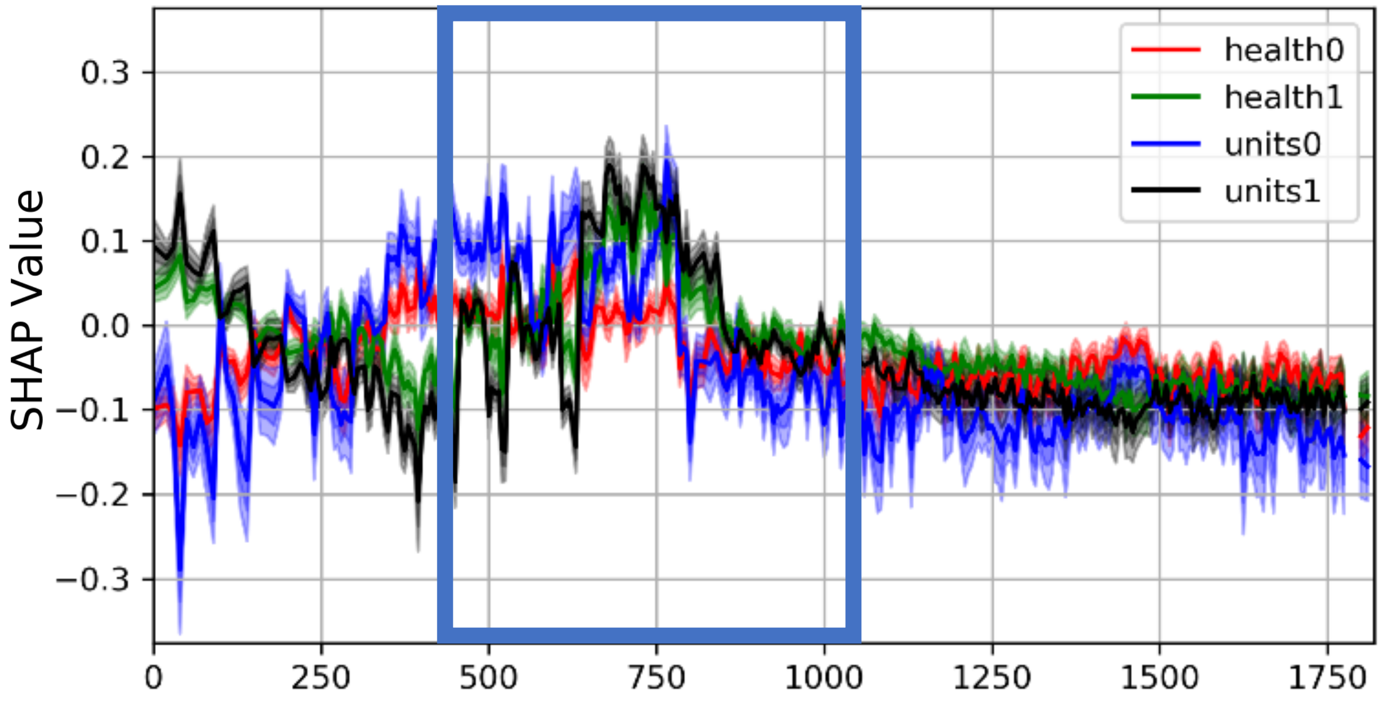


Examining model and SHAP outputs allows users to **identify where balance shifts**, and **what features seem most important** during these shifts.

SHAP Outputs: An Example (Cont…)



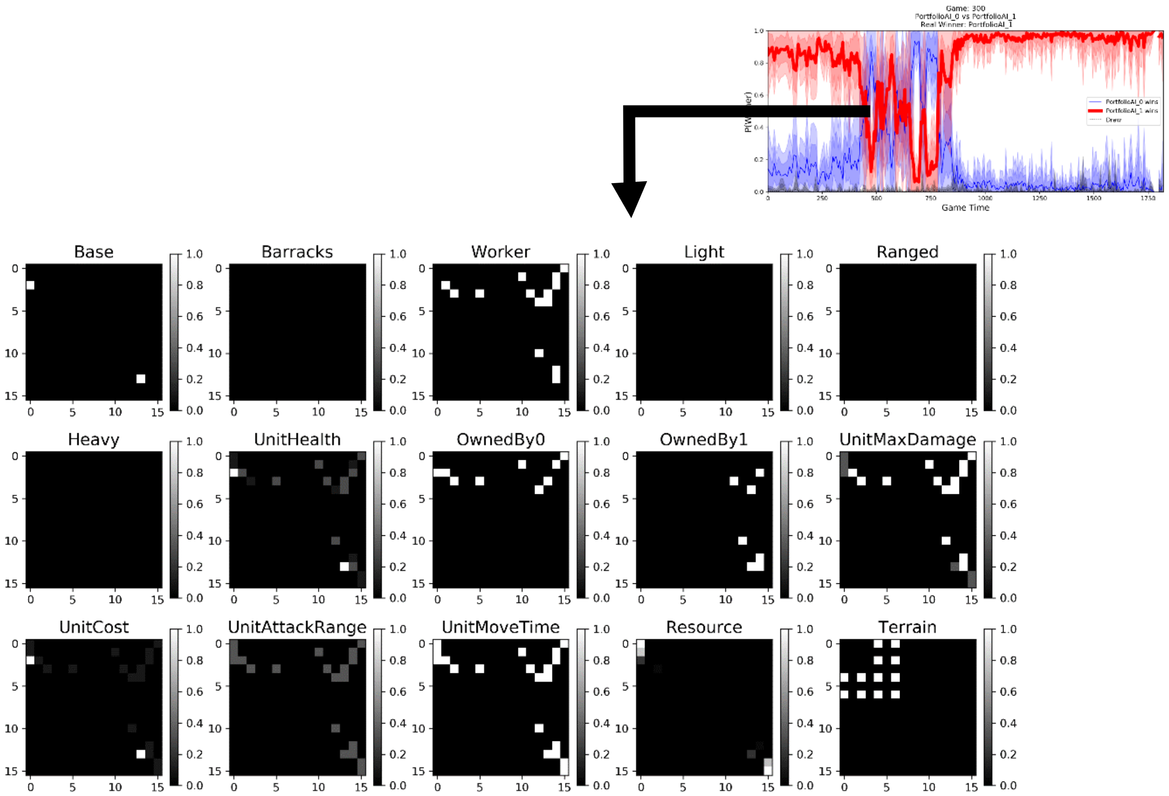
SHAP over Time, For Player 0 Winning



When Player 0 has more health and units on the field, the model becomes more confident that Player 0 will win.

Closing The Loop in Gamebreaking: Observations

- **Observation:** Significant factors in predicting Player 0's win:
  - ➢ The horizontal location of Player 0
  - ➢ Units of player 1
  - ➢ Health of player 1



- **Experiments:**

  At the current time:

  1. Move units of Player 0 horizontally towards Player 1's base
  2. Remove Player 1 unit health

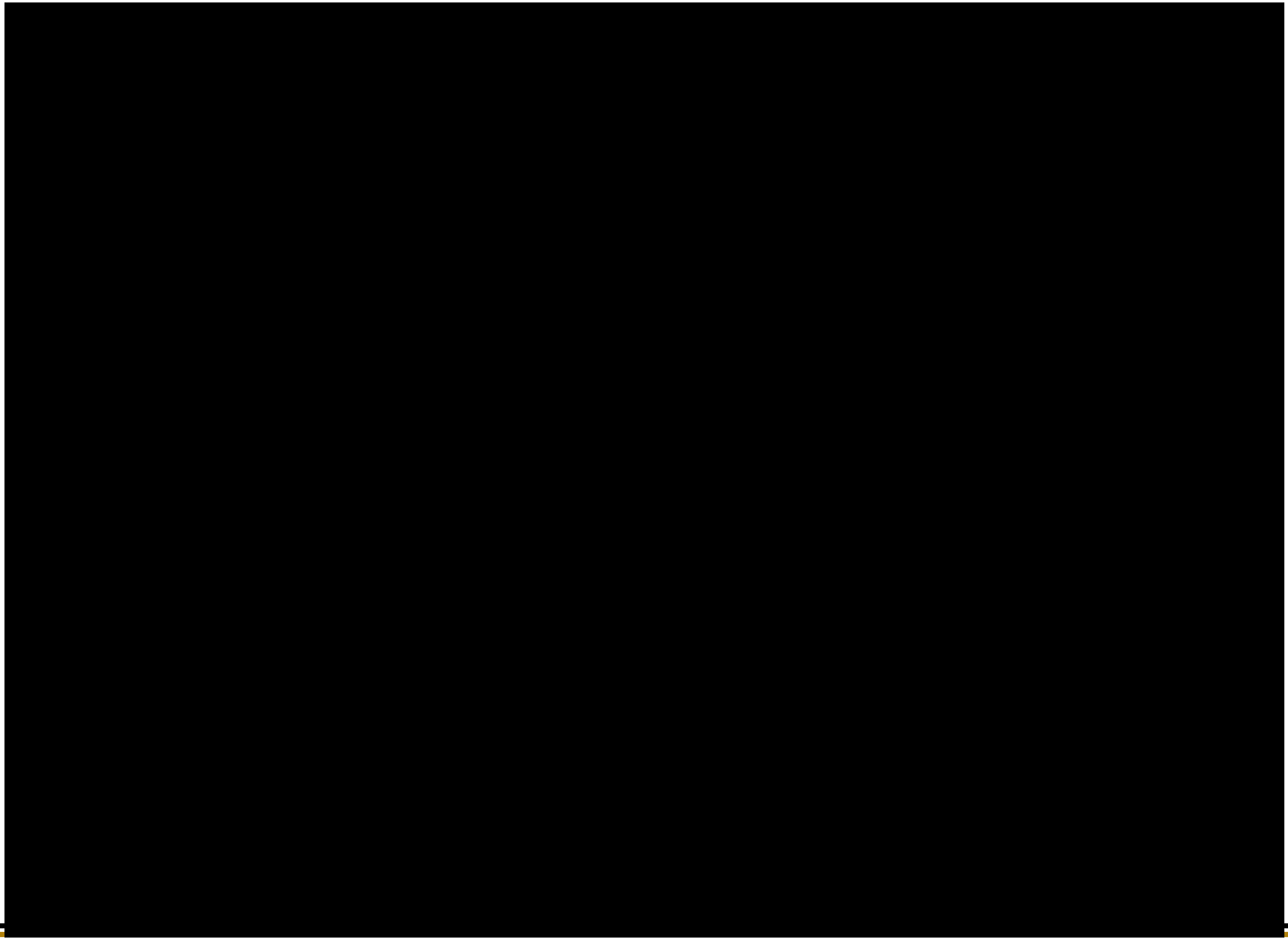
Closing The Loop in Gamebreaking: Video Demo

- **Observation:** Significant factors in predicting Player 0's win:
  - The horizontal location of Player 0
  - Units of player 1
  - Health of player 1

- **Experiments:**

  At the current time:

  1. Move units of Player 0 horizontally towards Player 1's base
  2. Remove Player 1 unit health



Closing the Loop with Mission Engineering and Game Breaking

Mosaic warfare and mission engineering broadly require:

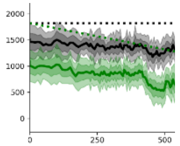
- **Actionable** and **dynamic** insights from the battlefield
- **Explanations of outputs** to better inform decision making

SHAP

Game Breaking Provides:

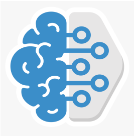
- Mechanism to understand and build balance equations from AI techniques (CNN model)
- Ability to identify what feature contribute to balance from XAI techniques (SHAP Model)

Future Directions



Incorporating Uncertainty Quantification and Optimal Learning

- Intelligently discover ways to refine data collection for *uncertain* regions



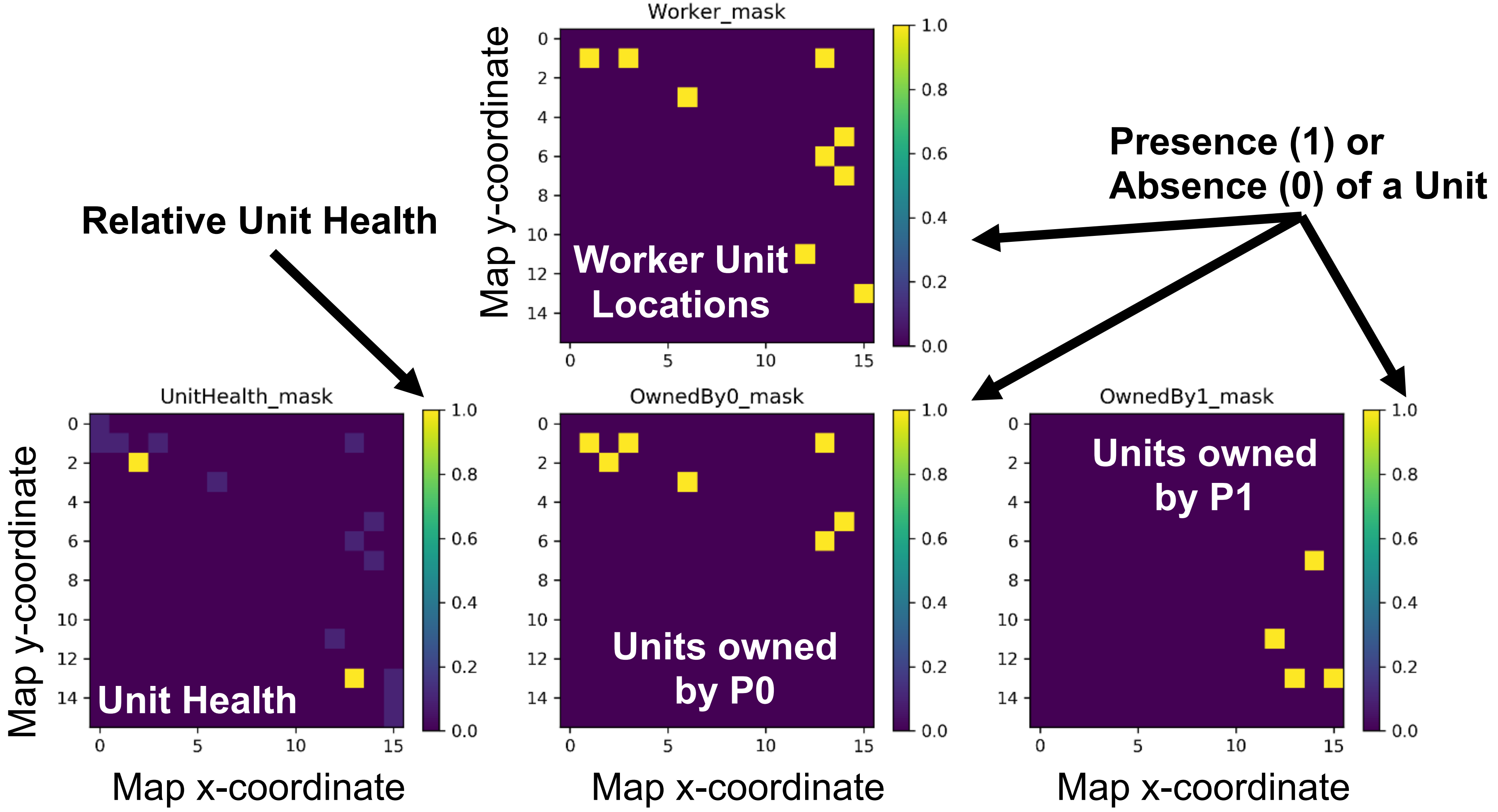
Scale to Complex RTS Games and Mosaic Scenarios

- Expand L2G framework implementation from MicroRTS to more complex scenarios

AI4SE **Generalize a path forward for AI and XAI for Systems and Mission Engineering**

Thank You

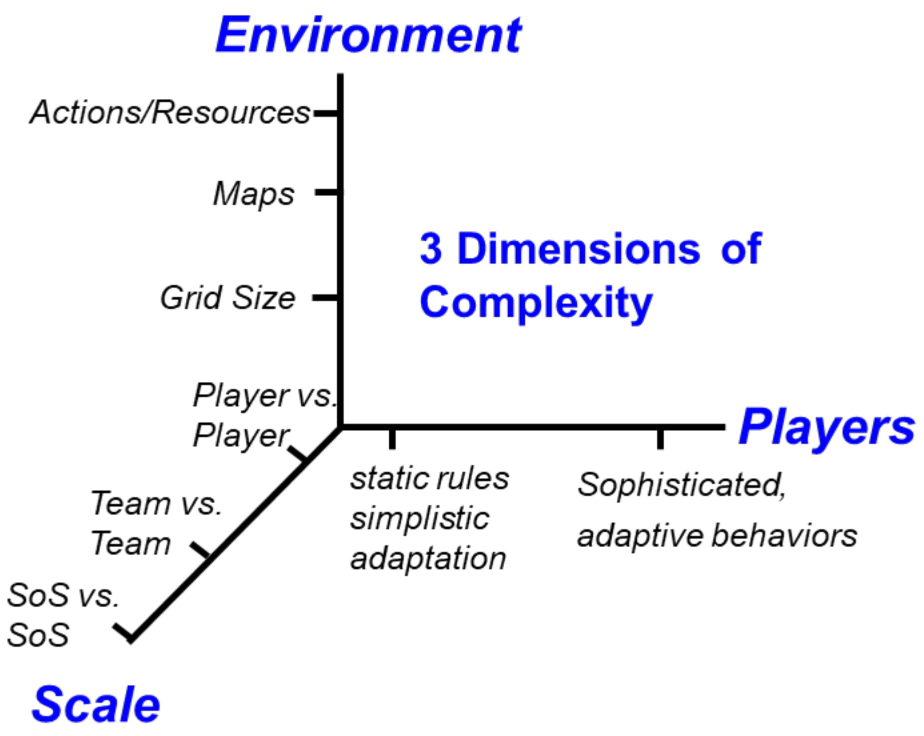
Input Data: Interpretation



Fundamental Challenges We Pursue (1): Complexity and Scalability

Challenge Description

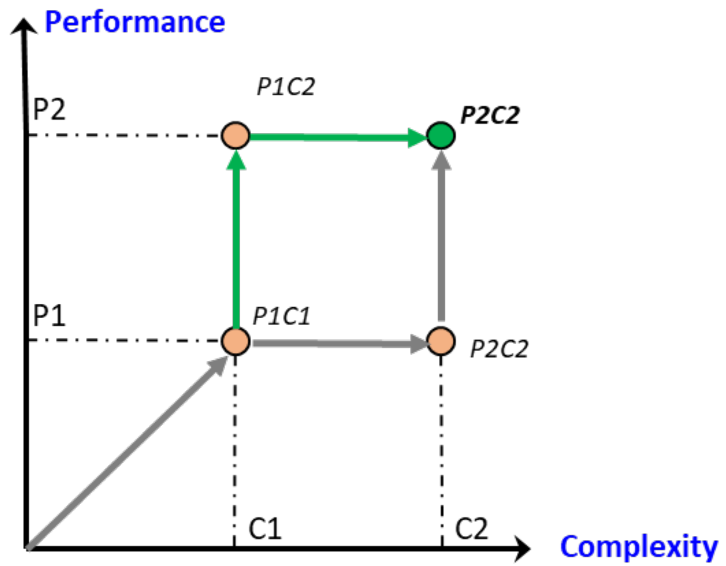
- How to learn amidst varying levels of diverse complexities?



- How does the nature of balance (and its measures) change with increased complexity?
  - Composite metrics and how to discover them
  - Are (optimal) balance and imbalance regions different?

Our Approach

- Sequential iterative refinement of design space
  - Start simple (e.g. use MicroRTS) and incrementally scale complexity towards real MOSAIC Warfare
  - CNN+SHAP to identify significant variables
  - Optimal Learning: Test and evaluate algorithms for detecting game balance, identifying optimal levers to imbalance
  - Create ***Performance vs. Complexity Map***



Fundamental Challenges We Pursue (2): Explainability and Complexity

Challenge Description

- How harder is "explainability" when complexity increases?
  - Even in MOSAIC warfare (a rapid, semi-automated composition and execution environment), doctrine will require explainability for decisions (perhaps after the fact, but nonetheless)
  - How to modulate predication algorithms according to degrees of human decision-maker confidence requirement
  - <u>Overall</u>: How to strike a balance between complexity, performance, and explainability for creating battlefield imbalance?

Our Approach

- Explainability is an independent dimension with performance and complexity in the Gamebreaker Layer
  - As we scale-up complexity, reexamine the sequence of methods and their interface in order to produce not only optimal prediction but also explainability