MassNE: Exploring Higher-Order Interactions with Marginal Effect for Massive Battle Outcome Prediction

Yin Gu
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence, Hefei, China
gy128@mail.ustc.edu.cn

Kai Zhang
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence, Hefei, China
sa517494@mail.ustc.edu.cn

Qi Liu*
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence, Hefei, China
Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, China
qiliuql@ustc.edu.cn

Xin Lin
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence, Hefei, China
linx@mail.ustc.edu.cn

Zhenya Huang
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence, Hefei, China
huangzhy@ustc.edu.cn

Enhong Chen
Anhui Province Key Laboratory of Big Data Analysis and Application, University of Science and Technology of China & State Key Laboratory of Cognitive Intelligence, Hefei, China
cheneh@ustc.edu.cn

ABSTRACT

In online games, predicting massive battle outcomes is a fundamental task of many applications, such as team optimization and tactical formulation. Existing works do not pay adequate attention to the massive battle. They either seek to evaluate individuals in isolation or mine simple pair-wise interactions between individuals, neither of which effectively captures the intricate interactions between massive units (e.g., individuals). Furthermore, as the team size increases, the phenomenon of diminishing marginal utility of units emerges. Such a diminishing pattern is rarely noticed in previous work, and how to capture it from data remains a challenge. To this end, we propose a novel **Mass**ive battle outcome predictor with margi**N**al **E**ffect modules, namely **MassNE**, which comprehensively incorporates individual effects, cooperation effects (i.e., intra-team interactions) and suppression effects (i.e., inter-team interactions) for predicting battle outcomes. Specifically, we design marginal effect modules to learn how units' marginal utility changing respect to their number, where the monotonicity assumption is applied to ensure rationality. In addition, we evaluate the current classical models and provide mathematical proofs that MassNE is able to generalize several earlier works in massive settings. Massive battle datasets generated by StarCraft II APIs are adopted to evaluate the performances of MassNE. Extensive experiments empirically demonstrate the effectiveness of MassNE , and MassNE can reveal reasonable cooperation effects, suppression effects, and marginal utilities of combat units from the data.

*Corresponding Author.

CCS CONCEPTS

• **Computing methodologies → Modeling methodologies**.

KEYWORDS

Marginal effect, Massive battle, Battle outcome prediction

ACM Reference Format:
Yin Gu, Kai Zhang, Qi Liu, Xin Lin, Zhenya Huang, and Enhong Chen. 2023. MassNE: Exploring Higher-Order Interactions with Marginal Effect for Massive Battle Outcome Prediction. In *Proceedings of the ACM Web Conference 2023 (WWW '23), April 30–May 04, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3543507.3583390

1 INTRODUCTION

A battle, usually involving two teams fighting against each other, is common in sports (e.g., football) and online games such as multiplayer online battle arena (MOBA) games. In this paper, we focus on massive battles, where exist different units of large quantity (e.g., Figure 1). As a complex form of battle, massive battles are ubiquitous in online strategy games, such as Warcraft III, Starcraft II, and EVE online. Predicting the outcome of massive battle is a fundamental and challenging task, which can provide a valuable reference for optimizing team configuration [5, 29, 46] before battle or tactical decision-making [7, 39] (e.g., attack or retreat) in battle.

In the literature, many efforts have been made to address the battle outcome prediction problem. Early studies [19, 21] aim to learn individual abilities (i.e., skill rating) from battle records. Despite being widely used, they omit interplays between individuals within a team. In their assumption, team members are independent of each other, and a team's ability is the sum of the team members' scores. To model interplays between individuals, neural network-based encoders are adopted to obtain team representations [8, 13, 29] for the

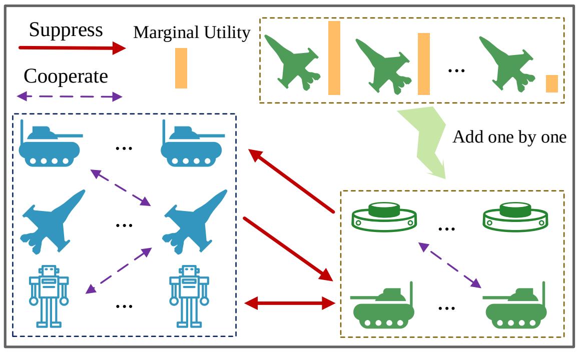


Figure 1: A massive battle from an online game, where the blue team and green team fight each other. Each team contains several unit types (e.g., tank, fighter, battle robot, and landmine) with varying numbers. As we keep adding units before the battle begins, their marginal utilities vary. Note that, in this paper, word *individual* denotes a combat unit in a team, regardless of its type. We use *squad* to refer to units of the same type in a team. For instance, a tank squad of the blue team represents all the tank units in the blue team.

outcome prediction. However, such work is not interpretable and it is hard to evaluate the contribution of each individual. Other works [15, 31] model pairwise interactions between individuals (e.g., intra-team cooperation effect or inter-team suppression effect). Although such works retain interpretability while considering interactions, they still have several downsides that hinder them from massive battle settings. 1) The number of interactions is the square of the number of individuals, when the team size becomes large, the efficiency cannot guarantee. 2) Interactions may not exist between all pairs of individuals, since the influence of each individual is not infinite, one does not interact with all others simultaneously.

The increase in team size complicates the combat scenario and presents unique challenges. As shown in Figure 1, two teams fight each other on the battlefield, where different unit types of varying numbers are involved. Units of the same team will cooperate, and units of different teams will confront each other (i.e., suppress or compete). Different unit types usually have distinct attack traits and defense traits. For example, tanks can cover ground units from enemy fire, but a few tanks can't cover hundreds of units at the same time. One landmine can incapacitate a tank (e.g., destroy the track and make the tank immovable), but one landmine cannot incapacitate an entire tank squad. The quantity of each squad makes the problem more tricky. How to consider the diverse higher-order interactions and the varying number of squads is an open problem.

The marginal effect is another important phenomenon that has been largely omitted by prior research but deserves more attention. As the number of a squad increases, the utility of this squad is usually not proportional to the number (i.e., the law of diminishing marginal utility [36]). Such a pattern is acknowledged as a marginal effect (i.e., the more unit, the smaller the benefit gains from the current increase). In our task, the term marginal refers to a small increase in the squad's quantity. Suppose there are two cases on the battlefield, in the first case, a team has no fighters, then 1 fighter is added; in the second case, a team has 100 fighters, then we add 1 fighter to it. The utility of the added fighter is different in the two cases, where the utility in the first case is huge (qualitative change), and the utility in the second case is small (quantitative change). Hence marginal effects exist in massive battles, and learning such effects is nontrivial for accurate predictions.

To overcome those unique challenges coupled with massive battles, we propose a **Mass**ive battle outcome predictor with margi**N**al **E**ffect modules (**MassNE**). We tackle the large-scale battle issue by treating the units of the same type (i.e., squad) as a whole. In particular, we considered the cooperation effect between allies and the suppression effect between enemies. We map each squad to multiple latent vectors and utilize DNNs as the interaction function to calculate the cooperation ability and suppression ability. Then, we develop marginal effect modules to capture the pattern of unit utility changing with the number. These modules are implemented by look-up tables, which can output the utility based on a given quantity. The marginal effect modules are learned in a data-driven manner, and they should adhere to the monotonicity assumption: a squad's utility should never decrease had one unit added. The main contributions of this work can be summarized as follows:

- We formalize the massive battle outcome prediction task and propose a new model MassNE, which efficiently learns cooperation and suppression effects between massive units.
- We notice the diminishing marginal utility of units in massive battles, and devise marginal effect modules to capture this pattern, which reveals each squad's marginal utility changing as a function of its quantity.
- Extensive experiments on three massive battle datasets verified the effectiveness and interpretability of MassNE. The datasets and codes[1] are public for future research.

2 RELATED WORK

2.1 Battle Outcome Prediction

Early studies considered one versus one (1V1) battles, famous algorithms such as Bradley-Terry [3], ELO [10] and Glicko [12] are widely deployed in chess and online games. Due to the emergence of team-based games, subsequent algorithms have been proposed for group battles. The classic work[19, 21] focuses on learning individual abilities (i.e., skill rating) from battle records, assuming that individual abilities are independent of each other. However, the interaction between individuals will affect the battle outcome, thus it should not be ignored (e.g., if two individuals cooperate well, they will have a higher chance to win). To model interactions between individuals, neural network-based methods [8, 13, 29] are utilized. They typically use a neural network to obtain team representations and then concatenate the representations of the two teams for the prediction. However, such work is not interpretable, making it hard to evaluate each individual's contribution to the battlefield. Other works [6, 15, 31, 40] model pairwise interactions between individuals (e.g., intra-team cooperation effect and inter-team suppression effect). For example, HOI [31] models the pairwise cooperation effect between teammates, and NeuralAC [15] uses neural networks to learn the cooperation effect and pairwise suppression (i.e., competition) effect between opponents. Although such models are interpretable, they have shortcomings that make them unsuitable for massive settings. They assume that each individual interacts, which is not true when the battlefield is large, since the influence of each individual is not infinite. Besides, as the team size increases, modeling the interaction between all individuals is less efficient.

[1]https://github.com/firepd/MassNE

Some researchers have manually designed combat models [7, 24, 41, 45, 46] for predicting combat winner, which requires additional unit information (e.g., attack damage and health point). Their main idea is, a unit with higher health points and attack damage is considered stronger. In contrast to those models, our model learns from the battle outcome and does not require additional information.

Another line of the work utilizes machine learning technology to incorporate rich in-game features for improving prediction accuracy [8, 13, 30, 32], or focuses on real-time battle winner prediction [49, 51]. In addition, Elo-MMR [9] estimate individual skill in a "free for all" setting of massive player contests. These works are orthogonal to ours.

2.2 Marginal Effect

As a common phenomenon, the marginal effect has been extensively studied in many other fields [14, 16, 27, 37, 43, 44], such as psychology and sociology. In particular, [14] analyzes the impact of the increase of high-status members on group effectiveness in business activities. [43] and [16] study how the addition of talented players changes the team performance in football or basketball. These works often rely on some predefined functions (e.g., quadratic function or log function) to fit data points, while our method learns from data and does not require predefined functions.

3 THE PROPOSED MODEL

In this section, we first formally define the massive battle outcome prediction task. Then, we give an overview of MassNE. After that, we detail the basic version of MassNE and its marginal effect modules. Finally, we demonstrate the generality of MassNE by comparing it with the existing models.

3.1 Problem Definition

Suppose there are N types of combat unit $\{1, 2, ..., N\}$ in total, the unit type combinations of a team is a subset of $\{1, 2, ..., N\}$. Let q_i ($q_i \geq 1$) denotes the quantity of unit type i in a given team. Meanwhile, there are H observable battles, each battle involves two teams T_A and T_B, and the outcomes of these H battles are denoted as $\{y_1, y_2, ..., y_H\}$. In this paper, we focus on the binary outcome prediction task, we assume that there is no draw. If T_A beat T_B in a battle $h \in [1, H]$, then $y_h = 1$, otherwise $y_h = 0$. Given an unseen battle between T_A and T_B, our goal is to predict the result $\hat{y} \in [0, 1]$.

3.2 Model Overview

When two teams are at battle, the one with the higher combat ability has a higher chance to win. Following previous work [4, 21], we assume each team has an underlying score indicating the team's capability, and we formulate the probability of team T_A defeating team T_B as:

$$\begin{aligned} P(A \text{ defeats } B) &= \frac{\exp(S_A)}{\exp(S_A) + \exp(S_B)}, \\ &= \sigma(S_A - S_B), \end{aligned} \tag{1}$$

where S_A and S_B are scores of T_A and T_B, representing the comprehensive ability of the two teams in a battle. σ is the sigmoid function. When two teams' scores are close, both teams have equal odds to win. When $S_A \gg S_B$, $P(A \text{ defeats } B) \to 1$, T_A will beat T_B with a high probability, and vice versa.

As mentioned above, there are multiple interactions among massive units [20]. Generally, if units' sole combat ability is strong or if several units cooperate well, the ability of the team can be improved. Besides, if units in T_A have more edges when against enemy units, then the winning probability of T_A can be increased. Therefore, in MassNE, the overall ability of the team consists of three parts: individual effects, cooperation effects, and suppression effects. Take T_A versus T_B as an example, we formulate T_A 's score as:

$$S_A = F_{\text{indi}}(T_A) + F_{\text{coop}}(T_A) + F_{\text{supp}}(T_A, T_B), \tag{2}$$

where the first term of S_A models individual effects. The second term $F_{\text{coop}}(T_A)$ models intra-team cooperation effects, and the third term $F_{\text{supp}}(T_A, T_B)$ captures inter-team suppression effects. Figure 2 shows the framework and main components of MassNE.

3.3 Basic MassNE

In this subsection, we illustrate the basic version of MassNE without marginal effect modules. Since the massive battle will generate massive interactions, we treat the units of the same type (i.e., squad) as a whole, which preserves the characteristic of each unit while reducing complexity.

3.3.1 Individual Effects. In our model, individual effects mean the ability of a squad that is independent of the other squad. To utilize the quantity information q_i, a straightforward idea is that the utility of a squad i is proportional to its number q_i. Hence we formulate the individual effect as:

$$F_{\text{indi}}(T_A) = \sum_{i \in T_A} q_i \cdot w_i, \tag{3}$$

where $w_i \geq 0$ is an individual ability, which is a trainable parameter. q_i is simply regarded as squad i's utility.

3.3.2 Cooperation Effects. If two squads perform well when they team up together, the cooperation ability between them should be high. Following [15, 31], we represent a unit type i's cooperation characteristics with a cooperation vector $\mathbf{v}_i \in \mathbb{R}^K$, because every unit type has distinct cooperation properties. The cooperation score between the two squads depends on their respective cooperation vectors and quantity:

$$F_{\text{coop}}(T_A) = \sum_{i \in T_A} \sum_{j \in T_A, i \neq j} q_i \cdot q_j \cdot f_1(\mathbf{v}_i \odot \mathbf{v}_j), \tag{4}$$

where $\odot$ denotes element-wise product, $\mathbf{v}$ is learnable parameter, f_1 indicates a MLP with non-linear activation function, which are capable of learning higher-order interactions [48] between teammates. The output of $f_1(\mathbf{v}_i \odot \mathbf{v}_j)$ is a scalar value, indicating the cooperation ability between i and j. The utilities of the two cooperative squads are q_i and q_j, which are their respective numbers. An increase in the number of either squad will increase the cooperation score between them. For clarity, $F_{\text{coop}}(T_A)$ is a overall cooperation score of T_A, and $q_i \cdot q_j \cdot f_1(\mathbf{v}_i \odot \mathbf{v}_j)$ represents cooperation score between i and j in T_A.

3.3.3 Suppression Effects. If squad a can suppress squad b (e.g., landmines counter tanks), the corresponding suppression ability from a to b should be high. Inspired by previous work [4, 15], we assume that every unit type has its strengths and weaknesses. We represent these two parts with two vectors, namely strength vector $\mathbf{p}_i \in \mathbb{R}^K$ and weakness vector $\mathbf{c}_i \in \mathbb{R}^K$. K is the embedding size. Further, a suppression ability depends on an attacker's strengths

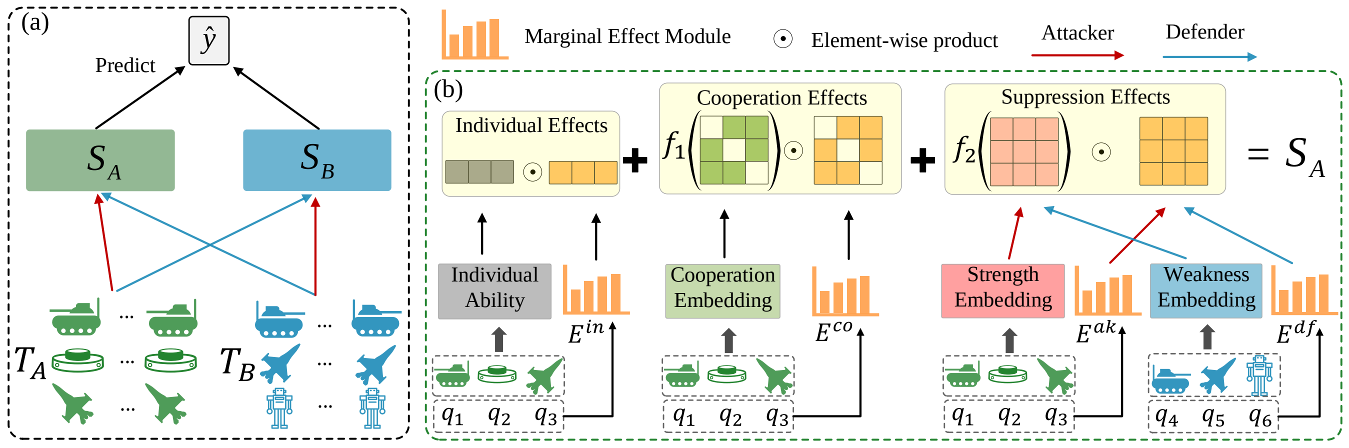


Figure 2: (a) MassNE framework. (b) The calculation of S_A.

and a defender's weaknesses. And the suppression score is related to the suppression ability and number of two teams:

$$F_{\text{supp}}(T_A, T_B) = \sum_{i \in T_A} \sum_{j \in T_B} q_i \cdot q_j \cdot f_2(\mathbf{p}_i \odot \mathbf{c}_j), \tag{5}$$

where $\mathbf{p}$ and $\mathbf{c}$ are learnable parameters, and f_2 is a MLP with non-linear activation function, which can model interactions between enemies. The output of $f_2(\mathbf{p}_i \odot \mathbf{c}_j)$ is a scalar value, indicating a suppression ability when i compete against j. If squad i has more advantage over squad j, then the corresponding $f_2(\mathbf{p}_i \odot \mathbf{c}_j)$ would be high and $f_2(\mathbf{p}_j \odot \mathbf{c}_i)$ would be low. We assume that the defender will not fight back when obtaining a suppression ability. For example, we do not consider the counterattack of the squads in T_B when calculating $F_{supp}(T_A, T_B)$, the attack of the squads in T_B will be modeled in $F_{supp}(T_B, T_A)$. It is intuitive that an increase in the number of attackers i increases the suppression score. The increase in the number of defenders will also increase the suppression score, because the defender j will not counterattack, increasing the defender will make the attacker play a greater role. Therefore, the utilities of the attacker i and the defender j are set to q_i and q_j.

In our implementation, f_1 and f_2 share the same network architecture, which is a two-layer neural network with ReLU activation functions. To guarantee the interpretability of MassNE (e.g., it does not make sense when cooperation and suppression scores are negative), the outputs of f_1 and f_2 are added with a ReLU function (i.e., $f_1(\cdot) \geq 0$ and $f_2(\cdot) \geq 0$).

3.4 MassNE with Marginal Effect Modules

This subsection shows how to enhance the basic MassNE with marginal effect modules. In the basic version, we simply set the squad's utility equal to q_i. However, the marginal effect should not be ignored in teamwork, especially when the team size is large. For example, when teammates cooperate, if two tanks can cover a teammate, then a third tank will not contribute much. When attacking the enemy, if ten cannonballs can destroy a tank, then an 11-th cannonball is possibly wasted. If one missile can destroy one battle robot, we only have five missiles while the enemy has ten battle robots, then the increase of enemy robots (i.e., defender) will not give us more advantage. In summary, a squad's utility does not increase linearly with the quantity. These patterns are hard to capture with hand-designed functions. Instead, we create learnable marginal effect modules that learn these effects from data.

Specifically, we use E to denote a marginal effect module, whose input is the number of a squad and outputs the squad's utility. Besides, individual effects, cooperation effects, and suppression effects do not share their marginal effect module. To summarize, we give the complete formulation of S_A:

$$\begin{aligned} S_A = &\sum_{i \in T_A} E_i^{in}(q_i) w_i \\ &+ \sum_{i \in T_A} \sum_{j \in T_A, i \neq j} E_i^{co}(q_i) \cdot E_j^{co}(q_j) \cdot f_1(\mathbf{v}_i \odot \mathbf{v}_j) \\ &+ \sum_{i \in T_A} \sum_{j \in T_B} E_i^{ak}(q_i) \cdot E_j^{df}(q_j) \cdot f_2(\mathbf{p}_i \odot \mathbf{c}_j). \end{aligned} \tag{6}$$

where E^{in} and E^{co} are marginal effect modules of individual effect and cooperation effect, respectively. E^{ak} and E^{df} are marginal effect modules of suppression effect, which can output the utilities of the attacker and the defender respectively. Due to units of different types having distinct characteristics (e.g., attack damage, armor), each unit type does not share its own E modules. Therefore, for given N unit types, MassNE has $4N$ independent E modules in total.

Now, we formally introduce our monotonicity assumption that marginal effect modules should adhere to: *If one unit is added to a team, the overall ability of the team should not decrease.* We assume each member in the team has the same goals (i.e., to win).

Specifically, in a massive battle, the utility of a squad should be monotonically increasing with respect to its quantity, and remaining non-negative. This assumption can be formulated as:

$$\begin{aligned} &E_i(q_i + 1) \geq E_i(q_i), \\ &E_i(q_i) \geq 0. \end{aligned} \tag{7}$$

We adopted a simple and feasible method to implement the E function, which is a look-up table (i.e., embedding layer) followed by a ReLU activation function. We add the ReLU function to ensure the output value (i.e., utility) is non-negative. Before training, each element in the look-up table is initialized to a positive value to prevent dead neurons caused by the ReLU.

Because constrained optimization in neural networks is tricky [1], we take a compromise approach to ensure monotonicity. To guarantee it (i.e., Equation (7)), we add a monotonicity constraint to the look-up table:

$$\mathcal{L}^m = \sum_{i=0}^{N} \sum_{k=0}^{MQ(i)-1} \left[E_i(k) - E_i(k+1)\right] \cdot I\left[E_i(k) > E_i(k+1)\right], \quad (8)$$

where I is the indicator function, $MQ(i)$ denotes the max quantity of i in the dataset. In Equation (8), If $E_i(k)$ is greater than $E_i(k+1)$, then $E_i(k) - E_i(k+1)$ will be considered in the loss, otherwise $E_i(k) - E_i(k+1)$ will be ignored.

E^{in}, E^{co}, E^{ak} and E^{df} have their corresponding monotonicity losses, i.e., $\mathcal{L}^{in}$, $\mathcal{L}^{co}$, $\mathcal{L}^{ak}$, and $\mathcal{L}^{df}$, respectively. The overall monotonicity loss can be expressed as:

$$\mathcal{L}_{mono} = \mathcal{L}^{in} + \mathcal{L}^{co} + \mathcal{L}^{ak} + \mathcal{L}^{df}. \quad (9)$$

It's worth noting that we do not impose additional constraints to ensure diminishing marginal utility. In experiments, we found that marginal effect modules can learn the diminishing pattern.

3.5 Training Strategy

Given H observed battles, let y_i denote the i-th battle outcome, $\hat{y}_i$ denote corresponding prediction , i.e., $P(A$ defeats $B)$. The loss function is cross entropy between model output $\hat{y}$ and true label y:

$$\mathcal{L}_{CE} = -\sum_{i=1}^{H}(y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i)). \quad (10)$$

The final loss is cross entropy loss plus monotonicity loss:

$$\mathcal{L} = \mathcal{L}_{CE} + \lambda \mathcal{L}_{mono}, \quad (11)$$

where λ is loss weight. In this way, we can learn MassNE by minimizing the loss function $\mathcal{L}$.

3.6 Generality of MassNE

In this subsection, we will review previous models, and demonstrate that previous work can be seen as special cases of MassNE. We first simplify the massive battle settings by reducing the number of each squad to 1. Since the quantity of each squad is 1, we can omit the marginal effect (i.e., E modules) of each squad, and the score of T_A is formulated as:

$$\begin{aligned} S_A = &\sum_{i \in T_A} w_i + \sum_{i \in T_A} \sum_{j \in T_A, i \neq j} f_1(\mathbf{v}_i \odot \mathbf{v}_j) \\ &+ \sum_{i \in T_A} \sum_{j \in T_B} f_2(\mathbf{p}_i \odot \mathbf{c}_j) \end{aligned} \quad (12)$$

Generalized Bradly-Terry. Generalized BT [21] assumes that individuals' abilities are independent of others, omitting interactions between individuals. By fixing f_1 and f_2's weights and bias to constant zero, we can get the generalized BT model, where the S_A is defined as:

$$S_A = \sum_{i \in T_A} w_i. \quad (13)$$

Higher Order Interactions. HOI [31] models pairwise cooperation effect by the inner product of two latent vectors $\mathbf{v}$, a team's score is defined as:

$$S_A = \sum_{i \in T_A} w_i + \sum_{i \in T_A} \sum_{j \in T_A, i \neq j} \mathbf{v}_i^T \mathbf{v}_j. \quad (14)$$

MLP can be seen as a generalization of the inner product operation [18], then if we let $f_2(\cdot) = 0$ (i.e., ignore the suppression effect), MassNE can recover HOI. **Blade-Chest-Inner.** Blade-Chest [4] is designed for 1v1 battle, which assumes that each player has an absolute ability w_i (i.e., individual effect), strength vector $\mathbf{p}_i$, and weakness vector $\mathbf{c}_i$. Blade-Chest considers the interaction between an opponent through the inner product of $\mathbf{p}_i$ and $\mathbf{c}_i$. In Blade-Chest-Inner model, take player a versus player b as an example, a's score is formalized as:

$$S_a = w_a + \mathbf{p}_a^T \mathbf{c}_b. \quad (15)$$

By limiting the team size of both sides to 1 (i.e., the cooperation effect disappears), our model can be reduced to the following formula, which can recover Blade-Chest model:

$$S_A = \sum_{i \in \{a\}} w_i + \sum_{i \in \{a\}} \sum_{j \in \{b\}} \mathbf{f}_2(p_i^T \mathbf{c}_j). \quad (16)$$

Basic NeuralAC. NeuralAC [15] learns the pairwise cooperative effects and pairwise suppression effects. When simplifying the quantity of each squad to 1, the T_A's score in NeuralAC is the same as in Equation (12). Therefore, MassNE can be regarded as an extended version of NeuralAC in massive battle settings.

4 EXPERIMENTS

4.1 Dataset Generation

Online games are an ideal testbed for simulating massive battles, which can provide unlimited battle data. StarCraft II is a famous online Real-Time Strategy (RTS) game that has attracted the attention of a large amount of AI communities in recent years [34, 39, 47]. In StarCraft II, Players need to collect resources, build buildings, and produce combat units to destroy enemies. The game contains three species for players to control: Terran, Protoss, and Zerg, and each of them have unique ground forces and air forces. We utilize sc2combatsim, an open-source StarCraft simulator [29] to simulate massive battles, which can generate two armies of specified quantity on the battlefield. A built-in AI will control two teams to attack, with a fixed combat policy. A battle ends when one team's army is eliminated, and the other team is considered to win.

We let the three species fight against each other, and we integrate most of the combat units of each species, including air forces. Note that some melee ground units (e.g., Zealot) and some air units (e.g., Viking Fighter) cannot attack each other. If this case happens at the end of the battle, we will filter this battle data.

For each battle, we randomly specify the maximum resource consumption of both sides. The amount of resources consumed by each squad in a team is determined by a Dirichlet distribution.

We generate three datasets on three different maps (i.e., terrains), namely **Plain**, **Corridor** and **Bush**. The basic statistics of all the datasets are summarized in Table 1. Plain terrain is relatively flat with no obstacles. Corridor terrain, in contrast, has a narrow corridor connecting two lands, which may blocks the movement of ground units. The Bush map has bushes on both sides that block the view of ground troops, but air forces are not affected by the bushes. The air forces would have a greater advantage on Corridor and Bush map than Plain.

4.2 Baseline Methods

- **Life Time Damage 2** (LTD2) [24]: LTD2 is a hand-craft model that assumes an individual's ability is determined by its attack damage, attack frequency, and hit point (i.e., health

Table 1: Statistics of the datasets.

Dataset	**Plain**	**Corridor**	**Bush**
Samples	34,540	33,104	34,494
#Unit types	39	39	39
Avg. team size	64.6	42.7	49.0
Max team size	797	479	461

point). In LTD2 , a team's ability is defined as:

$$S_A = \sum_{i \in T_A} q(i) \cdot \sqrt{HP(i)} \cdot DPF(i),$$

where $q(i)$ is number of i, $HP(i)$ denotes i's hit point and $DPF(i)$ denotes i's damage per frame (i.e., attack damage $\times$ attack frequency).

- **TS_Lanchester**[2] [46]: Another hand-craft model designed by experts, based on Lanchester's Square Law [26]. TS_Lanchester[2] takes into account the attack power and HP of the air forces and ground forces respectively. Therefore it considers the battle scenario at a fine-grain level.
- **Logistic Regression** (LR) [33]: A linear classifier with L2 regularization.
- **Generalized Bradly-Terry** (BT) [21]: Another linear model considers only individual effects.
- **TrueSkill** [19]: An algorithm based on probability graph, which is widely used in online team-based games for skill rating. It also only considers individual effects.
- **HOI** [31]: A factorization machines (FM) [38] based model that takes pairwise interactions of teammates into account.
- **NeuralAC** [15]: A neural network-based model that explicitly learns pairwise weighted-cooperation effects and weighted-suppression effects for battle prediction.
- **BattleNet** [28, 29]: A neural network-based model that uses multi-layer perceptrons to obtain representations of teams for final predictions.

4.3 Experimental Setup

For MassNE model, the dimension of embedding size (i.e., K) and the dimension of hidden layers are set to 20 and 64 [50], respectively. We initialize the parameter with Kaiming initialization [17]. Besides, Dropout [42] technique is also applied with the drop probability set to 0.2. The weight of monotonicity loss (i.e., λ) is set to 1.

For every dataset, we randomly divided samples into 80% for training, 10% for validating, and 10% for testing. Area Under ROC (AUC) [2] and Accuracy (Acc) are adopted the evaluation metrics. We choose Adam [23] as the optimizer, with 0.001 of learning rate and 0.0001 of weight decay coefficient [25]. Besides, the batch size is set to 64 for HOI, NeuralAC, BattleNet, and MassNE on all datasets.

LR and TrueSkill are implemented by open source python packages sklearn, trueskill[2], respectively. HOI, NeuralAC, BattleNet and our model are implemented by PyTorch package [35]. All experiments are conducted on a Linux server with an AMD Ryzen CPU and an RTX 3060 GPU.

[2]https://trueskill.org/

4.4 Experimental Results

Table 2 shows the experimental results of all methods on the massive battle outcome prediction task. We have the following observations. 1) Hand-craft models (i.e., LTD2 and TS_Lanchester[2]) perform poorly on this task. The main reason is that these models may oversimplify the battle process. They don't take into account the attack range of units or battlefield terrains, therefore don't perform well. 2) Trueskill, BT, and LR which consider only individual effects outperform rule-based methods because they are learning-based algorithms. 3) Although HOI and NeuralAC consider pairwise interactions between individuals, their performance is close to BT and Trueskill. It is may due to the assumptions made in their paper do not hold in the massive settings (in their experiments, the team size is usually five). When teams are large, not all individuals interact with each other. 4) BattleNet uses neural networks to obtain team representations for prediction, which fail to take into account the diminishing marginal utility of squads, resulting in suboptimal performance. Finally, MassNE outperforms all the other baselines on all datasets, indicating the effectiveness of our model.

4.5 Ablation Study

We further designed some model variants to verify the effectiveness and necessity of each design in MassNE. For fair comparisons, other settings remain unchanged.

- **basic**: Basic version of MassNE, the squad utility is not a learnable parameter, but the number itself, i.e., $E_i(q_i) = q_i$.
- **exist**: The number unity is 1 if a squad exists, and 0 otherwise, i.e., $E_i(q_i) = 1$ if $q_i >= 1$.
- **no-mono**: A variant of MassNE that ignores the monotonicity assumption, where the monotonicity loss is removed.
- **no-coop**: A variant that does not model the cooperation effect, i.e., assuming $f_1(\cdot) = 0$.
- **no-supp**: A variant that does not model the suppression effect, i.e., assuming $f_2(\cdot) = 0$.
- **no-indiv**: A variant that does not consider the individual effect, i.e., assuming $w_i = 0$.

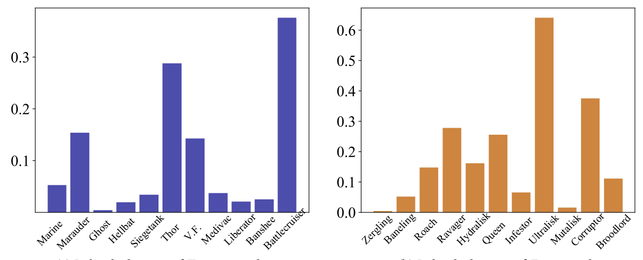
The ablation results are shown at the bottom of Table 2. We have several crucial conclusions. First, neither basic nor existing perform well. It proves that learning units' utility from the data (i.e., marginal effect modules) is better than manually specifying the utility. Second, the monotonicity loss is the key. We found when the monotonicity loss is removed, MassNE is prone to overfitting on the training set, which confirms the necessity of monotonicity loss. Third, all three effects (i.e., individual effect, cooperation effect, and suppression effect) do play a role, but the suppression effect is more important in Starcraft II.

4.6 Model Interpretability

To evaluate the interpretability of MassNE, i.e., whether the learned individual scores, cooperation scores, and suppression scores are reasonable, we display individual scores of each squad in Figure 3, cooperation scores between each squad in Figure 4 and suppression scores in Figure 5. These scores are obtained through MassNE trained on Corridor dataset. Please note that these scores depend not only on the type of units, but also on the number of squads. For example, the suppression score when i against j is calculated as $E_i^{ak}(q_i) \cdot E_j^{df}(q_j) \cdot f_2(\mathbf{p}_i \odot \mathbf{c}_j)$.

Table 2: Experimental results on massive battle outcome prediction task. The second-best methods are denoted with *. Those results are averaged over five independent runs, and standard deviations are shown in parentheses.

Model	Plain		Corridor		Bush	
	AUC	Acc	AUC	Acc	AUC	Acc
LTD2	0.6868 (0.0055)	0.6792 (0.0063)	0.6567 (0.0061)	0.6476 (0.0046)	0.6687 (0.0047)	0.6589 (0.0044)
TS_Lanchester[2]	0.7551 (0.0074)	0.7276 (0.0061)	0.6683 (0.0042)	0.6567 (0.0052)	0.7436 (0.0072)	0.7159 (0.0066)
BT	0.8931 (0.0036)	0.8159 (0.0014)	0.8646 (0.0055)	0.7843 (0.0062)	0.8887 (0.0061)	0.8033 (0.0073)
LR	0.8999 (0.0028)	0.8207 (0.0043)	0.8653 (0.0072)	0.7881 (0.0069)	0.8891 (0.0071)	0.8021 (0.0034)
TrueSkill	0.8954 (0.0028)	0.8113 (0.0034)	0.8276 (0.0129)	0.7458 (0.0089)	0.8699 (0.0073)	0.7869 (0.0049)
HOI	0.8943 (0.0053)	0.8129 (0.0063)	0.8616 (0.0051)	0.7818 (0.0071)	0.8894 (0.0072)	0.8044 (0.0075)
NeuralAC	0.8976 (0.0037)	0.8194 (0.0074)	0.8632 (0.0049)	0.7852 (0.0052)	0.8905 (0.006)	0.8085 (0.0065)
BattleNet	0.9234 (0.0027)	0.8245 (0.0039)	0.9133 (0.0021)	0.8137 (0.0049)	0.9074 (0.0046)	0.8117 (0.0074)
MassNE	**0.9489** (0.0024)	**0.8802** (0.0038)	**0.9460** (0.0035)	**0.8758** (0.0043)	**0.9412** (0.0021)	**0.8662** (0.0046)
basic	0.8968 (0.0025)	0.8145 (0.0054)	0.8633 (0.006)	0.7817 (0.0054)	0.8857 (0.0035)	0.8017 (0.0043)
exist	0.8703 (0.0037)	0.7949 (0.0045)	0.8907 (0.0073)	0.8196 (0.0086)	0.8592 (0.0049)	0.7819 (0.005)
no-mono	0.8737 (0.0055)	0.7947 (0.0043)	0.8882 (0.0065)	0.8140 (0.0092)	0.8916 (0.0043)	0.8068 (0.0034)
no-coop	0.9439 (0.0033)	0.8714 (0.0023)	0.9438* (0.0018)	0.8721* (0.0033)	0.9356 (0.0034)	0.8604 (0.0045)
no-supp	0.9030 (0.0056)	0.8213 (0.0064)	0.8835 (0.0087)	0.8023 (0.0068)	0.8976 (0.0067)	0.8131 (0.0087)
no-indiv	0.9463* (0.0017)	0.8759* (0.0055)	0.9427 (0.0038)	0.8688 (0.0043)	0.9376* (0.0038)	0.8612* (0.0053)



(a) Individual scores of Terran squads. (b) Individual scores of Zerg squads.

Figure 3: Individual scores. To get those scores, we assume that the number of each squad is one. Among the listed units, Thor is Terran's ultimate ground unit, and Battlecruiser is Terran's ultimate air unit. Ultralisk is Zerg's ultimate ground unit. Ultimate units mean that they are the strongest unit in combat, but also cost many resources to train.

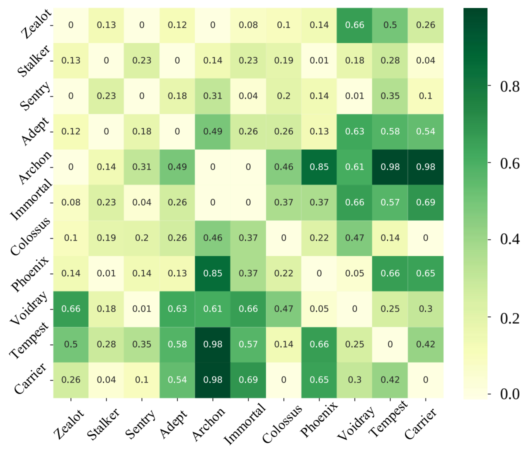


Figure 4: The cooperation scores of Protoss squads, i.e., $E_i^{co}(q_i) \cdot E_i^{co}(q_i) \cdot f_1(\mathbf{v}_i \odot \mathbf{v}_j)$. We set the number of two squads to 10 to calculate these scores. The diagonal is left blank since one squad will not interact with itself. Among them, Zealot, Stalker, Sentry, Adept, and Archon are ground units, and Phoenix, Voidray, Tempest, and Carrier are air units.

Individual Score. As shown in Figure 3, those ultimate units (e.g., Thor, Battlecruiser, and Ultralisk) get high scores, while the scores of junior units (e.g., Marine, Reaper, and Zergling) are quite low. This is consistent with the game scene.

Cooperation Score. As shown in Figure 4, We found that cooperation scores between ground units are low (i.e., Zealot and Immortal), while cooperation scores between ground units and air forces were high (i.e., Zealot and Voidray). One likely explanation may be that the ground units have a certain collision volume, which blocks the units behind them on the narrow corridor, making them hard to cooperate. The air unit is not hindered by the terrain, and hence air forces cooperate well with the ground forces.

Suppression Score. In the Corridor terrain, we noticed that the Siegetank counters the Zerg ground units (i.e., Roach and Ravager) because when the tank is Sieged, it has the longest attack range and high damage. However, tanks cannot attack air units, so tanks get a low score against the enemy air force (e.g., Corruptor and Broodlord). In addition, as the ultimate air unit, the Battlecruiser can attack both air and ground units, so the suppression score of the Battlecruiser is generally high.

Through the above analysis, we can infer that MassNE indeed learned meaningful relationships between each squad. It is worth pointing out that our model is capable to discover such complex relationships merely based on battle outcomes, without consuming any prior knowledge (e.g., attack range and HP) [3].

[3]For more details on StarCraft II units, interested readers can refer to the following website: https://liquipedia.net/starcraft2/Units_(Legacy_of_the_Void)

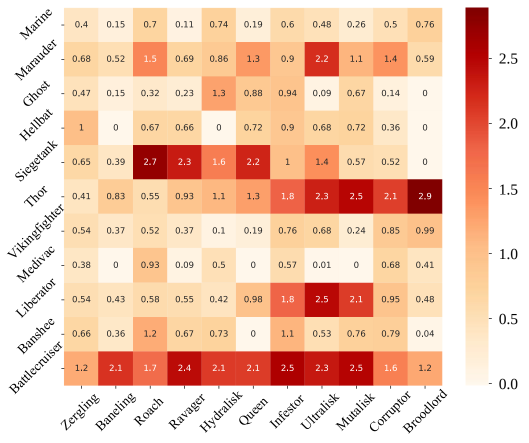


Figure 5: The suppression scores of Terran squads fight against Zerg squads. We set the number of two squads to 10 to obtain these scores. In particular, Siegetanks are ground units and cannot attack air units (e.g., Mutalisk, Corruptor, and Broodlord).

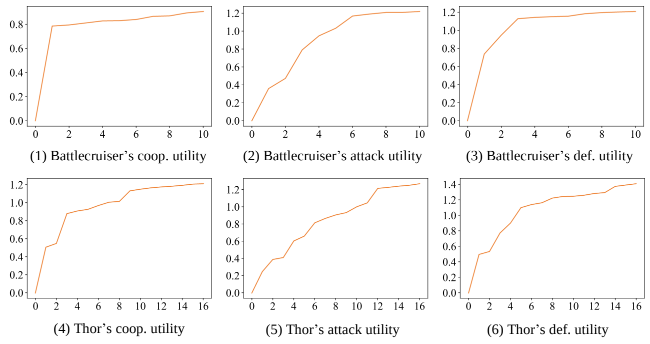


Figure 6: Curve of squad utility as a function of quantity. The vertical axis represents the quantity of each squad.

4.7 Diminishing Marginal Utility

To observe the learned utility change pattern, we obtained the learned parameters of marginal effect modules in Corridor and plot the curve of utility as a function of the squad number. We choose Thor and Battlecruiser and draw their utility curves (i.e., E^{co}, E^{ak} and E^{df}), the results are displayed in Figure 6.

We can observe that in general, the marginal utility will continue to decrease as the number increases (marginal utility is $E(k+1) - E(k)$). We speculate that there are several reasons for this phenomenon: when attacking, too many combat units will lead to wasted firepower; when cooperating, too many ground units will block each other, reducing the offensive efficiency. Therefore, utility is not proportional to quantity. In addition, different squads have different curve shapes, and it is hard to describe such patterns with manually designed functions.

4.8 Impact of Marginal Effects

To further explore the impact of the marginal effect on the winning probability, we plotted the probability of the two teams as a function of team size, and the results are shown in Figure 7. For demonstration, let's assume that T_A has only Ultralisk and T_B has only Thor. Ultralisks and Thors are the ultimate units of Zerg and Protoss respectively. Ultralisks are melee units, while Thors are ranged units. From Figure 7, we observe that the win rate increases as the number of units increases (keeping the opponent team the same). This is reasonable, since increasing the number will increase utility. In addition, when the team size is small (e.g., 1V1 or 3V3), the winning odds of the two teams are close; but as the team size increases (e.g., 6V6 or 8V8), Thor's winning odds are higher. A plausible explanation is that the marginal utility of Ultralisks decays faster than Thor on the corridor. Because only three Ultralisks can attack Thor simultaneously, and almost all Thors can attack Ultralisks at the same time, due to their long attack range.

Based on this table, we can make a pre-battle plan. For example, if we want to beat 6 Ultralisks, we need to deploy 5 Thors, and if we want to defeat 8 Ultralisks, we need to deploy 6 Thors.

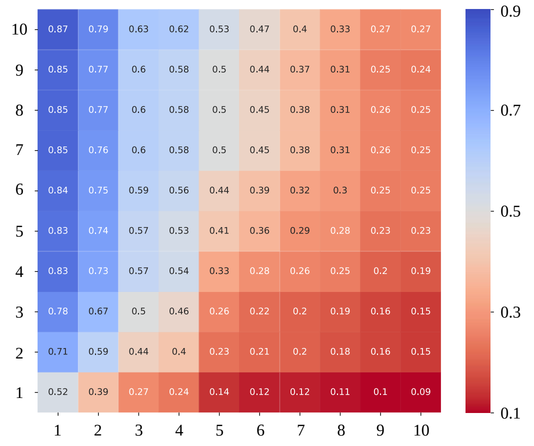


Figure 7: Winning probability. The vertical axis represents the number of Ultralisks, and the horizontal axis represents the number of Thors.

5 CONCLUSION AND FUTURE WORK

In this paper, we proposed a novel model named MassNE for massive battle outcome prediction, which considers higher-order interactions between each squad. The marginal effect modules of MassNE model the marginal utility of each squad. Experiments on three StarCraft II datasets demonstrated that MassNE outperforms state-of-the-art methods and confirmed the correctness of the monotonicity hypothesis. In addition, MassNE exhibits good interpretability and can reveal cooperation relationships and suppression relationships between each squad. The marginal effect modules can discover diminishing marginal utility patterns through end-to-end learning. Finally, we have demonstrated that MassNE could be seen as the generalization of several previous models.

We currently adopt StarCraft-II as the emulator, which is a relatively complex game. It has a variety of units (each unit has different attack damage, attack range, armor, and skills) and diverse terrain. Therefore, StarCraft covers a considerable number of online games owing to its complexity. Nevertheless, we acknowledge that other domains are more challenging to model, mainly due to many uncertainties or unobservable information. We think extending MassNE to other domains is a potential future direction, which can draw on the theory of swarm intelligence [22] or group dynamics [11].

ACKNOWLEDGMENTS

This research was partially supported by grant from the National Natural Science Foundation of China (Grant No. 61922073), and the University Synergy Innovation Program of Anhui Province (GXXT-2021-002).

REFERENCES

[1] Dimitri P Bertsekas. 2014. *Constrained optimization and Lagrange multiplier methods*. Academic press.

[2] Andrew P Bradley. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern recognition* 30, 7 (1997), 1145–1159.

[3] Ralph Allan Bradley and Milton E Terry. 1952. Rank analysis of incomplete block designs: I. The method of paired comparisons. *Biometrika* 39, 3/4 (1952), 324–345.

[4] Shuo Chen and Thorsten Joachims. 2016. Modeling intransitivity in matchup and comparison data. In *Proceedings of WSDM*. 227–236.

[5] Zhengxing Chen, Truong-Huy D Nguyen, Yuyu Xu, Christopher Amato, Seth Cooper, Yizhou Sun, and Magy Seif El-Nasr. 2018. The art of drafting: a team-oriented hero recommendation system for multiplayer online battle arena games. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 200–208.

[6] Zhengxing Chen, Yuyu Xu, Truong-Huy D Nguyen, Yizhou Sun, and Magy Seif El-Nasr. 2018. Modeling game avatar synergy and opposition through embedding in multiplayer online battle arena games. *arXiv preprint arXiv:1803.10402* (2018).

[7] David Churchill, Abdallah Saffidine, and Michael Buro. 2012. Fast heuristic search for RTS game combat scenarios. In *Eighth Artificial Intelligence and Interactive Digital Entertainment Conference*.

[8] Olivier Delalleau, Emile Contal, Eric Thibodeau-Laufer, Raul Chandias Ferrari, Yoshua Bengio, and Frank Zhang. 2012. Beyond skill rating: Advanced matchmaking in ghost recon online. *IEEE Transactions on Computational Intelligence and AI in Games* 4, 3 (2012), 167–177.

[9] Aram Ebtekar and Paul Liu. 2021. Elo-MMR: A Rating System for Massive Multiplayer Competitions. In *Proceedings of the Web Conference 2021*. 1772–1784.

[10] Arpad E Elo. 1978. *The rating of chessplayers, past and present*. Arco Pub.

[11] Donelson R Forsyth. 2018. *Group dynamics*. Cengage Learning.

[12] Mark E Glickman. 1999. Parameter estimation in large dynamic paired comparison experiments. *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 48, 3 (1999), 377–394.

[13] Linxia Gong, Xiaochuan Feng, Dezhi Ye, Hao Li, Runze Wu, Jianrong Tao, Changjie Fan, and Peng Cui. 2020. OptMatch: Optimized Matchmaking via Modeling the High-Order Interactions on the Arena. In *SIGKDD*. 2300–2310.

[14] Boris Groysberg, Jeffrey T Polzer, and Hillary Anger Elfenbein. 2011. Too many cooks spoil the broth: How high-status individuals decrease group effectiveness. *Organization Science* 22, 3 (2011), 722–737.

[15] Yin Gu, Qi Liu, Kai Zhang, Zhenya Huang, Runze Wu, and Jianrong Tao. 2021. Neuralac: Learning cooperation and competition effects for match outcome prediction. In *AAAI*, Vol. 35. 4072–4080.

[16] Bartosz Gula, Nemanja Vaci, Rainer W Alexandrowicz, and Merim Bilalic. 2021. Never too much-The benefit of talent to team performance in the National Basketball Association: Comment on Swaab, Schaerer, Anicich, Ronay, and Galinsky (2014). *Psychological science* 32, 2 (2021), 301–304.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*. 1026–1034.

[18] Xiangnan He and Tat-Seng Chua. 2017. Neural factorization machines for sparse predictive analytics. In *ACM SIGIR*. 355–364.

[19] Ralf Herbrich, Tom Minka, and Thore Graepel. 2007. TrueSkill: a Bayesian skill rating system. In *NeurIPS*. 569–576.

[20] Jie Huang, Qi Liu, Fei Wang, Zhenya Huang, Songtao Fang, Runze Wu, Enhong Chen, Yu Su, and Shijin Wang. 2021. Group-Level Cognitive Diagnosis: A Multi-Task Learning Perspective. In *2021 IEEE International Conference on Data Mining (ICDM)*. IEEE, 210–219.

[21] Tzu-Kuo Huang, Chih-Jen Lin, and Ruby C Weng. 2008. Ranking individuals by group comparisons. *JMLR* 9, Oct (2008), 2187–2216.

[22] James Kennedy. 2006. *Swarm intelligence*. Springer.

[23] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[24] Alexander Kovarsky and Michael Buro. 2005. Heuristic search applied to abstract combat games. In *Conference of the Canadian Society for Computational Studies of Intelligence*. Springer, 66–78.

[25] Anders Krogh and John A Hertz. 1992. A simple weight decay can improve generalization. In *NeurIPS*. 950–957.

[26] Frederick William Lanchester. 1916. *Aircraft in warfare: The dawn of the fourth arm*. Constable limited.

[27] Richard Layard, Guy Mayraz, and Stephen Nickell. 2008. The marginal utility of income. *Journal of Public Economics* 92, 8-9 (2008), 1846–1857.

[28] Donghyeon Lee, Man-Je Kim, and Chang Wook Ahn. 2020. Battlenet: Capturing advantageous battlefield in RTS games (student abstract). In *AAAI*, Vol. 34. 13849–13850.

[29] Donghyeon Lee, Man-Je Kim, and Chang Wook Ahn. 2021. Predicting combat outcomes and optimizing armies in StarCraft II by deep learning. *Expert Systems with Applications* 185 (2021), 115592.

[30] Hojoon Lee, Dongyoon Hwang, Hyunseung Kim, Byungkun Lee, and Jaegul Choo. 2022. DraftRec: Personalized Draft Recommendation for Winning in Multi-Player Online Battle Arena Games. In *ACM Web Conference*. 3428–3439.

[31] Yao Li, Minhao Cheng, Kevin Fujii, Fushing Hsieh, and Cho-Jui Hsieh. 2018. Learning from group comparisons: exploiting higher order interactions. In *NeurIPS*. 4981–4990.

[32] Tom Minka, Ryan Cleven, and Yordan Zaykov. 2018. Trueskill 2: An improved bayesian skill rating system. *Tech. Rep.* (2018).

[33] Jordan Michael I Ng, Andrew Y. 2002. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In *NeurIPS*. 841–848.

[34] Santiago Ontanón, Gabriel Synnaeve, Alberto Uriarte, Florian Richoux, David Churchill, and Mike Preuss. 2013. A survey of real-time strategy game AI research and competition in StarCraft. *IEEE Transactions on Computational Intelligence and AI in games* 5, 4 (2013), 293–311.

[35] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. 2019. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*. 8026–8037.

[36] Robert S Pindyck and Daniel L Rubinfeld. 2014. *Microeconomics*. Pearson Education.

[37] Dongning Ren, Olga Stavrova, and Wen Wei Loh. 2022. Nonlinear effect of social interaction quantity on psychological well-being: Diminishing returns or inverted U? *Journal of Personality and Social Psychology* 122, 6 (2022), 1056.

[38] Steffen Rendle. 2010. Factorization machines. In *ICDM*. IEEE, 995–1000.

[39] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder De Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. 2019. The starcraft multi-agent challenge. *arXiv preprint arXiv:1902.04043* (2019).

[40] Aleksandr Semenov, Peter Romov, Sergey Korolev, Daniil Yashkov, and Kirill Neklyudov. 2016. Performance of machine learning algorithms in predicting game outcome from drafts in Dota 2. In *International Conference on Analysis of Images, Social Networks and Texts*. Springer, 26–37.

[41] Dennis Soemers. 2014. Tactical planning using MCTS in the game of StarCraft. *Master's thesis, Maastricht University* (2014).

[42] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* 15, 1 (2014), 1929–1958.

[43] Roderick I Swaab, Michael Schaerer, Eric M Anicich, Richard Ronay, and Adam D Galinsky. 2014. The too-much-talent effect: Team interdependence determines when more talent is too much or not enough. *Psychological Science* 25, 8 (2014), 1581–1591.

[44] Richard Tay. 2008. Marginal effect of increasing ageing drivers on injury crashes. *Accident Analysis & Prevention* 40, 6 (2008), 2065–2068.

[45] Alberto Uriarte and Santiago Ontanón. 2015. Automatic learning of combat models for RTS games. In *Eleventh artificial intelligence and interactive digital entertainment conference*.

[46] Alberto Uriarte and Santiago Ontanón. 2017. Combat models for RTS games. *IEEE TOG* 10, 1 (2017), 29–41.

[47] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. 2019. Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature* 575, 7782 (2019), 350–354.

[48] Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang, Yu Yin, Shijin Wang, and Yu Su. 2022. NeuralCD: A General Framework for Cognitive Diagnosis. *IEEE Transactions on Knowledge and Data Engineering* (2022).

[49] Zelong Yang, Zhufeng Pan, Yan Wang, Deng Cai, Shuming Shi, Shao-Lun Huang, Wei Bi, and Xiaojiang Liu. 2022. Interpretable Real-Time Win Prediction for Honor of Kings–a Popular Mobile MOBA Esport. *IEEE TOG* (2022).

[50] Kai Zhang, Hao Qian, Qing Cui, Qi Liu, Longfei Li, Jun Zhou, Jianhui Ma, and Enhong Chen. 2021. Multi-interactive attention network for fine-grained feature learning in ctr prediction. In *Proceedings of the 14th ACM international conference on web search and data mining*. 984–992.

[51] Chuang Zhao, Hongke Zhao, Yong Ge, Runze Wu, and Xudong Shen. 2022. Winning Tracker: A New Model for Real-time Winning Prediction in MOBA Games. In *ACM Web Conference*. 3387–3395.