Predicting Victories in Video Games - IEEE BigData 2021 Cup Report

Maciej Matraszek*†, Andrzej Janusz*†, Maciej Świechowski†, Dominik Ślęzak*†
*Institute of Informatics, University of Warsaw, Poland
†QED Software, Poland
Contact Email: andrzej.janusz@qed.pl

2021 IEEE International Conference on Big Data (Big Data) | 978-1-6654-3902-2/21/$31.00 ©2021 IEEE | DOI: 10.1109/BigData52589.2021.9671650

***Abstract*—We summarize the results of IEEE BigData 2021 Cup: Predicting Victories in Video Games - a data mining challenge organized at the KnowledgePit platform in association with the IEEE BigData 2021 conference. We describe the competition task, as well as the data acquisition and preprocessing steps. We also provide a brief overview of the top-performing solutions submitted by participants. Finally, we present results of the post-competition data analysis, in which we consider the similarity of solutions submitted by various teams in terms of their errors on the test data. We conclude this analysis by demonstrating a method for constructing an ensemble of submitted solutions. Such an ensemble performs better than any of the individual solutions submitted during the competition.**

***Index Terms*—Data mining competitions; video game analytics; ML in video games;**

I. Introduction

The constant growth of the video game industry and remarkable advances in machine learning (ML) stimulate research in this domain. Game developers increasingly often rely on modern ML techniques to design their games and provide a better gaming experience to players [1]. Among many different applications of ML in GameDev, the most profound ones are related to designing *better* (e.g., interesting, challenging, human-like) AI for computer opponents [2] and to game analytics [3]. Moreover, game frameworks are often employed as research or test environments in various "real-world" multi-agent problems. Such an approach is referred to as *gamification* [4].

In both of those applications areas, i.e, analytics and creating AI agents, there is a common task that can be tackled by using modern ML techniques, i.e., the prediction of chances for a particular player winning the game, either based on game logs or historical gaming performance.

There are many ways in which such an information can be utilized. For example, an AI bot capable of predicting the winner can be geared towards making stronger actions or weaker ones depending on the chosen difficulty level. In simulation-based techniques to AI such as Monte Carlo Tree Search (MCTS) [5], the win-chance predictor can act as a heuristic evaluation function that approximate the game outcomes. Normally, the MCTS algorithm performs quasi-random full game simulations to obtain samples – results of the game – from a given position. Using a win-chance prediction model enables to perform search only to a certain depth in the game tree [6].

From the game analytics perspective, we can analyze which aspects of the game affect the win-chances the most. First of all, we can test whether the game is balanced [7], i.e, there are no overpowered strategies or elements that make the game unfair for players not using them. Second of all, we can analyze which of the elements featured in the game work the best, what kind of strategies are the most effective, and finally - we can profile players in terms of their skills.

This paper presents a report on *IEEE Big Data 2021 Cup* competition that concerned predicting victories in the game called *Tactical Troops: Anthracite Shift*. We introduce the game and explain how the data was gathered for it in the next Section. Section III is devoted to the description of the competition setting. In Section IV, we provide the report that includes analysis of the submitted entries. Finally, the last Section contains conclusions.

II. Game and Input Data

A. Game Description

Tactical Troops: Anthracite Shift is a 2D game of tactics and dexterity available on the Steam platform[1]. The game features a single-player campaign mode, but in the context of game analytics and our data mining competition, we focus on the multi-player mode only. Herein, two players – either human or AI – face each other in a game played on one of the 24 available maps as shown in Figure 1. Players control four units at the start but as the game progresses, units may be eliminated.

The game is turn-based, but unlike in many classical turn-based board games, players may take a lot of actions in a single turn. Each unit has a pool of 100 action points, which are resources for making actions. For example, shooting using a particular weapon may cost 60 action points. The movement is continuous without any grid and its cost in action points is proportional to the distance travelled. Possible actions for each unit include

[1]https://s.team/a/1266890

This work was co-funded by the Smart Growth Operational Programme 2014-2020, financed by the European Regional Development Fund under a GameINN project POIR.01.02.00-00-0207/20, operated by The National Centre for Research and Development.

Fig. 1: Screenshot from a match in Tactical Troops: Anthracite Shift.

- Shooting with a chosen weapon. There are over 30 weapons to choose from. Moreover, some weapons have more than one mode of firing, i.e., single-shot and series.
- Reloading a weapon
- Using and/or activating gadgets (30+ available) such as shields, mines, grenades, stimulants etc.
- Continuous movement
- Switching to an overwatch stance, i.e., sacrificing a turn for an auto-shot when the enemy gets in range in their turn.

The large number of available weapons and gadgets is mitigated by the fact that each unit may bring only a subset of 2 weapons and 3 gadgets for a particular match. Before a match, players also choose types of units they bring to the match. The unit types, which are *assault*, *heavy*, *support* and *scout*, differ by mobility, health and competence in using specific gadgets.

The turn ends when the player decides to pass, e.g., because no unit has action points left, or the time for a turn is up.

There are two ways of winning the game. The first victory condition applies to every map and consists in simply killing all enemy units. The second one is map-dependent and it may be either *domination* or *devastator*. In the domination mode, the first player to control the majority (i.e., 2 out of 3) of special regions, called control points, for a whole turn wins the game. Players control these regions by having more units within their range than the enemy. If a map does not feature control points, it must contain special stationary elements called generators, which are characteristic to the *devastator* mode. Each player has several generators of their own and the first player to destroy the enemy generators wins the game. The existence of the second victory condition encourages proactive game-play and prevents players from using very defensive strategies.

Game environment is dynamic to some extent. There are various types of objects that can be interacted with, e.g., movable and exploding crates or doors that can be opened/closed. Teleportation is one of the unique features of the game. Maps may contain paired teleports and any non-stationary object such as units, bullets, grenades or crates can be teleported. Teleportation preserves the momentum of the objects that come in and come out.

B. Data Acquisition

The primary source of data for game analytics [3] are logs captured from historical games. The logs usually contain information about the starting setup for a given match, e.g., chosen map, types of units and their equipment and the important events that occurred in specific time during the game. Such empirical logs are often complete (contain all evidence to reconstruct the game) but too raw and, therefore, not yet ready to be processed by machine learning algorithms. For the data formats provided to the participants of the competition, please refer to Section III-A.

We have acquired over 50 thousands of logs played in various configurations: (1) human vs. human, (2) human vs. AI bot and (3) AI bot vs. AI bot. We use the terms "computer players", "bots" and "AI bots/players" interchangeably. The last group was the largest and it contributed significantly more logs than the other two to the overall data set. The portion of games performed by humans was very low due to the fact that the competition had been prepared before the game release. Games played by computer players are the fastest to perform and nowadays, companies start to use such automated AI-driven methods to test games in addition to performing manual testing. Although, bots may play games in a different fashion to humans, they may as well explore vast portion of the game space that would require a lot of time and thousands of human testers, which would be prohibitively expensive. Not only can they find bugs but also provide data to statistical analysis, e.g., if the game is properly balanced. The more intelligent AI bots become, the more useful they can be in the testing process.

That is yet another reason (in addition to providing interesting opponents) behind developing human-like computer players. For this task, we used the *Grail* [2] library for AI in games. Our bots are constructed in a hierarchical fashion using concepts from multi-agent systems. The underlying technique for decision making combines Monte Carlo Tree Search (MCTS) and Utility AI algorithms. The details are described in [8]. MCTS has been the staple method in the realm of combinatorial games such as Go [9], Chess [10] or Hex [11]. It is particularly suitable for turn-based games played on discrete boards. However, in *Tactical Troops* we have continuous movement and a lot of choices that increase the combinatorial complexity for any search-based or exploration-based algorithms. Therefore, we use various heuristics to narrow down the choices to the most sensible ones as well as a fast Utility AI method that analyzes the game on a granular strategic level [12]. The output of Utility AI processing is prioritization of choices in the form of orders for each unit. Such orders are passed to the MCTS algorithms as the goal function. The main benefit of that is that MCTS does not need to simulate games to the end to obtain the estimated score (as in the classic algorithm). The simulations terminate when the order is executed or cannot be executed anymore.

For the task of generating logs from automated games, we developed the so-called *Arena*, which is effectively a server that does matchmaking between various bots' parameterizations, orchestrates matches between them and gathers the resulting logs. We made sure that there is some diversity between bots that mainly stems from the fact that various bots use different sets of equipment.

III. Competition Outline

IEEE BigData 2021 Cup: Predicting Victories in Video Games was organized using the KnowledgePit platform[2] by QED Software. It was held between April 30, 2021 and October 3, 2021. The competition task was to predict the winners in the Tactical Troops: Anthracite Shift games at a given time during the match. In this section, we provide details regarding the competition problem and summarize the results.

A. The data provided to participants

The collected logs were filtered to eliminate draws, aberrant games (e.g. having an idle player or no shots fired), rare game versions or otherwise malformed game logs. For each of the remaining 58658 matches a random cutoff turn was selected with a slight preference of mid-game situations. The contestants did not know how long the actual game took, what simulated an online prediction scenario. The data was then split into a training set featuring 38658 games and a test set with 20000 games divided into a preliminary and final part of, respectively, 2000 and 18000 games. The splits were random with the exception of games with human players, which all were pre-selected into the training set. The split between preliminary and final part of the test set was not public and during the competition the scores on the preliminary set were available to contestants and displayed on the leaderboard.

The gameplay data was provided in four formats, differing in the granularity of game descriptions: *truncated logs format*, *flattened logs format*, *tabular format*, and *screenshots*. Moreover, since the game was still in development during the collection of logs, we provided *metadata* information for each game version.

Truncated logs format: This data collection contained slightly modified JSON log files generated by the Tactical Troops: Anthracite Shift game. Each file represented one match between two players with timeline of status change such as unit movement, bullets firing and trajectory, damage received, control point capture etc. These files did not contain the full game description but they were truncated after a number of turns, yet, in essence, augmenting the data set with previous turns was possible. Some summary statistics of a match were also removed. Other game metadata such as player IDs were anonymized.

Flattened logs format: This representation format was more granular than the *tabular Format*, yet simpler to parse than the *truncated logs format*. In contrast to the latter, this format represented only the current state of the game, when the prediction is to be made, with all the state updates collected in a single dictionary for each entity.

[2]https://knowledgepit.ml/predicting-victories-in-video-games/

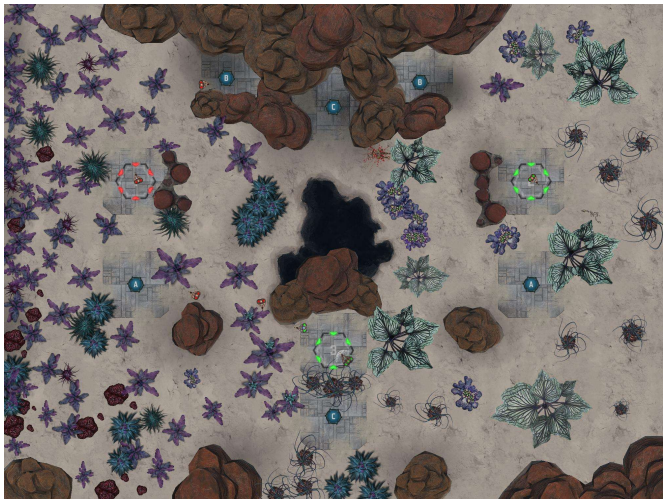


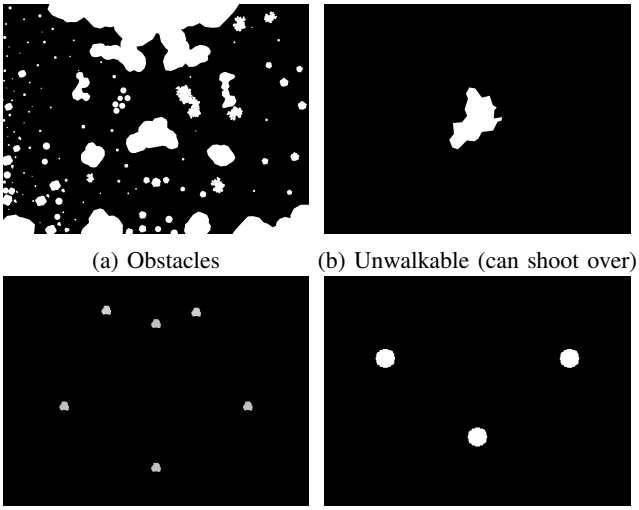
Fig. 2: An example *screenshot* in the provided data set. Two regions are captured by the green team and one by the red.

Tabular format: The simplest representation of played games consisted of an aggregated description of the situation on a map like status of troops health or map name. Each row in those tables corresponds to a single game with an identifier that allowed integration of multiple representations. In total 196 mixed categorical and numerical features were provided. Games in this representation were stored as a CSV file each for training and tested set.

Screenshots: As a complementary format, *screenshots* of the game state at the decision time were provided. The pictures, as shown in Figure 2, were generated by employing the Tactical Troops Replay application, and provided all the spatial information that an observer may see during a multiplayer match. When combined with binary layer masks found in the *metadata*, features such as actual shortest-path distance, line-of-sight or unit visibility under cover could be computed directly.

Metadata: Since the game underwent balancing changes over time, *metadata* for each version was provided. This includes various statistics of the plethora of featured weapons, gadgets, unit types etc. Moreover, the aforementioned map layers were published as binary-mask images representing obstacles and impassable regions, rooftops, as well as teleporters, control points and generators location (cf. Fig. 3).

In order to explain the granulation of the provided data, let us consider the most important statistic of a game unit: health points (*hp*). The *hp* value of a unit at the decision time is present in the *tabular format*, and from *metadata* one may retrieve the maximal *hp* of such unit type. From the *flattened logs format*, the current position of units and exploding barrels may be read, thus allowing for detection of proximity to explosives or, when combined with map layers *metadata*, whether enemies have a clear shooting line. Furthermore, the



(a) Obstacles (b) Unwalkable (can shoot over)

(c) Teleporters location (d) Control points

Fig. 3: Binary mask of map layers.

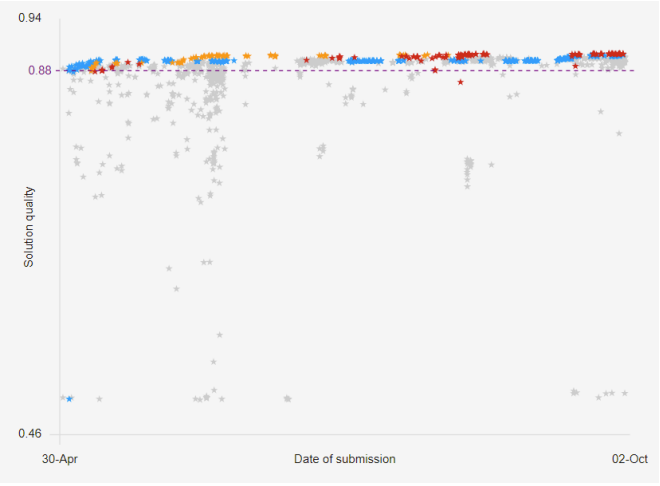


Fig. 4: A distribution of scores in time. Colours of the spots indicate submissions from the top three teams. Orange spots correspond to *tks*, blue to *henryvu*, and red to *Cyan* team.

truncated logs format contains information when exactly the unit lost its *hp*, which unit shot the bullets, whether *hp* was restored previously, i.e. how the match progressed over time.

B. Evaluation and baseline models

In the competition, predicting the winner of a Tactical Troops game is a binary classification problem, as exactly two teams compete with no draws allowed. The solutions were evaluated using Area Under Curve (AUC) metric, since it abstracts from a concrete threshold value and provides more fine-grained evaluation of the submissions.

Our baseline model utilised only the *tabular log format* and was selected using an auto-ml approach basing on the cross-validation score on the training dataset. More specifically, `lazy-predict`[3] library was employed, which tests multiple `scikit-learn` and scikit-learn-compatible models. The selected model utilized XGBoost [13] with no feature engineering whatsoever. Only the categorical features were either one-hot encoded (when low cardinality) or ordinal-encoded otherwise by the library.

We also experimented previously with training a dedicated neural network that was hierarchically building representation of game concepts. For instance, a representation of a squad is a function of a set of units, while vectorization of a unit depends on its hp, set of weapons, position on the map etc. Embeddings of categorical features (e.g. map name) were learned directly. The design of the network covered symmetries in the data, as well as incorporated convolutional parts for processing the current state of the game map (both local to a troop and global) reconstructed from the logs. The architecture was inspired by the input processing network in OpenAI Five [14]. Regrettably, even though all possible turn cutoff points were present in the training data, the model performance was considerably inferior to the score of the baseline XGBoost model.

[3]https://github.com/shankarpandala/lazypredict

TABLE I: The final AUC values and number of submissions from top-ranked teams. The last row shows the result obtained by the baseline model in this challenge, which we constructed as a reference for participants.

team name	rank	number of submissions	final result (AUC)
Cyan	1	138	0.8997
henryvu	2	388	0.8980
tks	3	71	0.8978
Dymitr	4	363	0.8963
GameOver	5	216	0.8950
PKAW	6	32	0.8908
Mathurin ACHE	7	33	0.8900
...	...	...	...
baseline	11	7	0.8822
...	...	...	...

C. Summary of the competition results

The competition lasted for over five months, from April 30, 2021, to October 03, 2021. In total, it attracted 125 teams from 23 countries, of which 43 submitted at least one solution to the public leaderboard. Overall, during the course of the competition, participants submitted 1629 solutions (1596 were correctly formatted and evaluated by the system). Additionally, 21 teams shared a description of their solution in a form of a short report, which was a necessary condition for being included in the final leaderboard. We present the final AUC scores obtained by the top-performing teams in Table I. Comparing to our baseline model, the final results of the top five teams were over 1% better than the baseline, which we consider to be a very good result. Figure 4 shows the distribution of scores obtained by contestants in time.

A common characteristic of the most successful solutions was that they all were heavily relying on extensive feature engineering to construct good representations of the game states. For example, the winners [15] defined over 3100 features

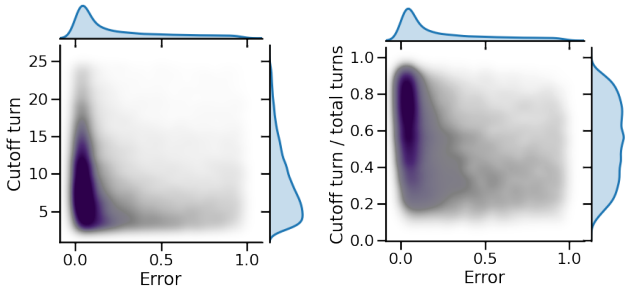
divided into eight groups (i.e., Metadata, Unit, Weapon, Gadget, Mode, Player, Flattened Logs, and Truncated Logs), and proposed a feature selection algorithm, named GRFE, whose aim was to maximize the performance of the LightGBM [16]. Similarly, the second team [17] also focused on constructing a large set of features, and then applied three different feature selection strategies (i.e., greedy forward search, greedy backward search, and SHAP value-based feature importance rankings) to fine-tune the final data representations. These representations were later used as the input to LightGBM in order to construct an ensemble of three prediction models - one general, and two specific to particular game modes. The third team utilized a different implementation of the boosted decision trees model, i.e., XGBoost [13]. However, as in the case of the first two teams, they perform a four-stage feature engineering process before the construction of the final model. More details about the three best solutions in the competition can be found in papers [15], [17], [18].

IV. Post-competition Analysis

The submitted predictions evaluated with the AUC metric may be interpreted as either probabilities (assuming cross-entropy optimization) of winning the game by team 1 or as a ranking (discarding the exact prediction values) of games. We analyzed the submission data with both of these perspectives in mind. First of all, we did not notice any team overfitting to the preliminary score, what is a common situation in data mining competitions [19]. On the contrary – the final score was predominantly underestimated by $0.005 - 0.01$. Despite this fact, the final ranking is nearly identical to the preliminary one, with only a few contestants swapping places.

A. Error analysis

We attempted to find the source of errors, in terms of probabilities, made by models developed by the contestants. We expected that aleatoric uncertainty would be an issue at the beginning of the game: while the combination of units, weapons and gadgets (so-called *loadout*) influences the probability of winning a match, it should be hard to predict the winner if the game is well balanced and the actions of players are relevant to its outcome. It may be, in fact, observed on Figure 5a that initial turns have a wide distribution of errors of the predicted probabilities by the winner solution. The analysis of relationship between model error and how far in the game the turn was sampled, expose a dramatic increase of the prediction accuracy at around halfway though the match (cf. Fig. 5b). Moreover, we noticed a considerable reduction of models' performance on human players: i.e the wining submission has AUC=0.9002 on subset of the final test set containing only AI vs. AI matches and drops to AUC=0.8576, when human players are present. One possible explanation might be that those games were played on a different game version. Conversely, one may attribute the epistemic uncertainty to the fact that human players were not present in the training set and that repeatable characteristics of the AI bots do not transfer well to human play.



(a) By cutoff turn number. (b) By the cutoff-total turn ratio.

Fig. 5: Heatmaps of errors made by the top solution. The colormap is two-slope iso-propotional: purple has approx. 5x the slope of gray.

Yet another approach to error analysis is to compare errors made by different contestants on the same match id. Our intuition is that, because the competitors' models are loosely correlated and employ independent feature engineering, agreement or disagreement of predicted scores may reveal unusual cases in the data. We sampled the matches that had the highest prediction error among many submissions. Manual inspection of the game replays confirmed that a human expert would confidently vote the same as the models – in those matches an unexpected change occurred, leading to victory of the weakened player. Automatically locating such situations in the large amount of gameplay logs could, for instance, help in debugging and improving AI players to eliminate poor decisions made by the agents. What is also evident in Figure 6 is that some contestants were submitting binary predictions instead of scores.

B. Submissions clustering

We wanted to verify whether we could meaningfully compare submissions, possibly enabling information discovery from processing the distance matrix. One of the use cases could be detection of prohibited cooperation between teams: sharing models, feature generation process etc. On the other hand, finding high-scoring, yet dissimilar submissions could be employed in finding interesting approaches to the problem or building well-generalizing ensembles.

If we consider the euclidean distance between submissions viewed as a vector of predicted probabilities of the positive class for each match (or, equivalently, the errors of predictions) the resulting value could be interpreted as L2 loss of a regression problem with one of the solutions as labels. However, as it is apparent from Figure 6, the distribution of errors vary a lot from submission to submission, what could be perhaps attributed to distinct optimized losses and score normalization. In order to better reflect submissions similarity in terms of the AUC score, we may deal solely with the ranking induced by the submission values and employ non-parametric statistics. If we treat each ranking as a vector, then the euclidean distance between such vectors is roughly equivalent to Spearman's ρ correlation coefficient and taxicab distance to Kendall's τ –

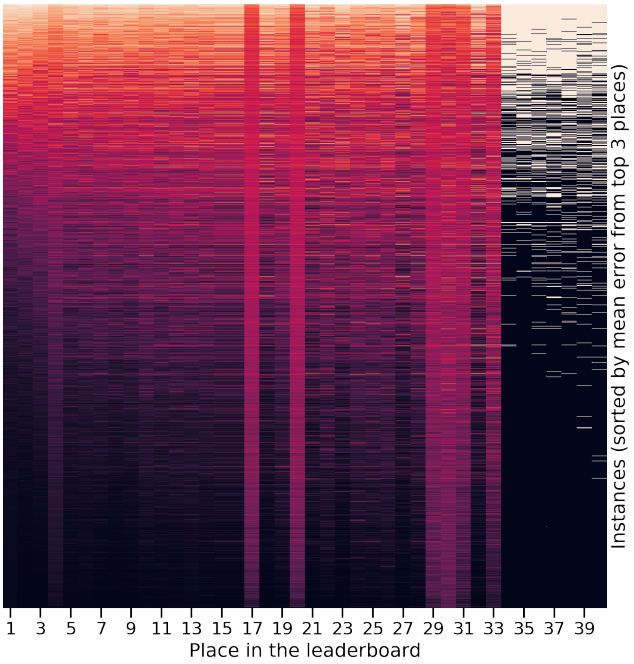


Fig. 6: Prediction errors made by best submission per team. Each row represents a test case and the color the prediction error in $[0, 1]$ (black-white) range. Different ranges of color in columns is the result of different predicted scores distributions. The rightmost submissions had binary predictions.

it is a sum of position displacements between the rankings. Therefore, we hypothesize that distance of errors strongly reflects the final model dissimilarity while the distance of rankings abstracts from the loss function and it may capture the similarities in data processing, even when completely different models were trained.

In the following analysis, we present the results when applying Minkowski distance of order 1.5 to the ranking vectors. The p value of 1.5 was chosen because it experimentally resulted in the most promising distance: the value of 1 (taxicab) exaggerates local perturbations of the ranking (i.e. due to a very close probability values) while with $p = 2$ (euclidean) a single outlier overshadows otherwise similar rankings. All submissions from the competition embedded into 2D space by t-SNE with the aforementioned metric are shown in Figure 7. In the embedding, multiple submissions of the same team tend to be grouped together indicating minor refinements in the predicted ranking. Of the top-scoring teams, only submissions from *henryvu* (pink) show significant progression in the ranking order. On the other hand, submissions from team *Dymitr* (orange) form four clusters, two of which consist solely of the solutions of a single member of the team.

Further inspection of clusters consisting of various teams unveils their characteristics. The top-left located cluster of positions 34-40 covers submissions with binary predictions, as opposed to required continuous scores for AUC metric (c.f. Fig. 6). Another intriguing cluster, of which the baseline solution is the very center, is the middle one incorporating mid-rank participants. In fact, an analysis of their reports, confirms that the predictions were made with gradient boosted trees (possibly selected by an auto-ml approach) with only rudimentary handling of categorical variables.

TABLE II: Final AUC values and the number of base models used in ensembles. The first row corresponds to the result of the competition winner. *The average top 5* and *weighted average top 5* are the results obtained for the simple averaging and weighted averaging, respectively. The last four rows show results of the investigated solution stacking method.

ensemble	final result (AUC)	no. base models
Cyan - competition winner	0.8997	1
average top 5	0.9003	5
weighted average top 5	0.9004	5
LASSO reg. (opt. λ) best	0.9018	4
LASSO reg. (max 5) best	0.9016	5
LASSO reg. (opt. λ) all	0.9020	10
LASSO reg. (max 5) all	0.9018	5

C. Ensembles of submissions

We also considered the problem of finding the optimal ensemble of solutions submitted by contestants. To this end, we adapt the technique of prediction model stacking [20]. In this approach, the mix of base models (contestants' submissions) is constructed in a supervised fashion, i.e., the predictions of base models on a validation set are used as training examples for a different classifier whose aim is to combine them into the final predictions.

In our post-competition analysis, we investigated two scenarios for constructing such a classifier. In the first one, we used all correctly formatted submissions, and in the second scenario, we restrained the submissions to those with the best score in the preliminary evaluation from each team. In both cases, we used a logistic regression model with LASSO regularization. We trained it on predictions for the preliminary evaluation set used in the completion, thus the final evaluation scores of individual solutions and our ensembles were computed on exactly the same set of test instances (i.e., the final evaluation set). We tuned the regularization parameter λ of our model using the cross-validation technique. For our comparison, we chose two λ values - the one for which the validation loss was lowest, and the smallest one for which the resulting ensemble was composed of at most five base models. We present evaluation results of the obtained ensembles in Table II. As a reference, we also provide the results obtained using simple averaging of best solutions from top five teams, and results of weighted averaging in which the weights correspond to preliminary scores of the top five solutions.

Interestingly, the model corresponding to the optimal λ value, trained on the best solutions from each team, had non-zero coefficients only for four teams, i.e., *Cyan* (1), *henryvu* (2), *tks* (3), and *GameOver* (5). These coefficients were 0.9894, 0.8342, 1.2672, and -0.2486, respectively. It means that the model from *tks* team had the highest impact

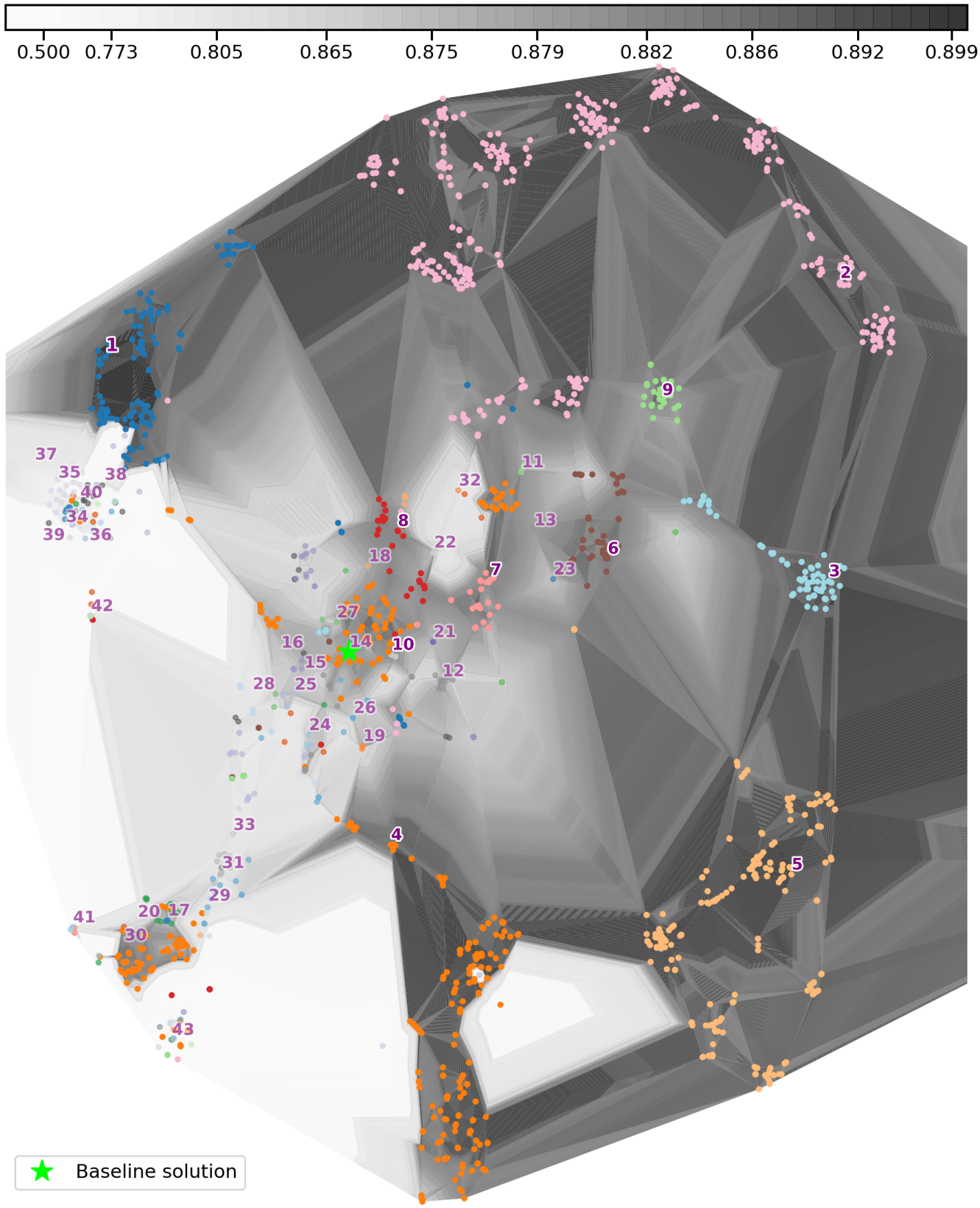


Fig. 7: TSNE embedding of all submissions represented as ranking vectors with Minkowski distance of order 1.5. Each submission is represented with a dot colored by the team. The highest-scoring submission of each team is marked with their position in the ranking. AUC scores are represented by shades in the background with ranking-based color scale normalization: i.e. the middle point of the color bar represents the position half through the ranking. Thus, by looking on the background of any dot we may estimate the position in the final competition ranking of a team that would send just this single submission.

on the final predictions. Moreover, the negative coefficient of the solution from *GameOver* means that this team either assumed a different ordering of target classes or focused on predicting ranks instead of win probabilities. The ensemble which achieved the highest AUC value was trained on all solutions. It had non-zero coefficients for ten solutions from five teams – two solutions from *Cyan*, two from *henryvu*, two from *tks*, one from *GameOver*, and three from *Dymitr*. As previously, the highest coefficient was corresponding to the best solution from *tks* team and the coefficient of *GameOver*'s solution was negative.

V. Summary

In this paper we summarized the Predicting Victories in Video Games data mining challenge organized as a part of IEEE BigData 2021 Cup. We started by discussing the competition data acquisition and preprocessing steps, and followed by explaining the various data modalities that we made available for the contestants. We briefly described the baseline models and characterized the best solutions from the top three teams. Finally, we performed an exploratory analysis of post-competition data and demonstrated how predictive models stacking technique can be used to construct an ensemble that outperforms all of the solutions submitted during the competition.

The experience gained throughout this competition including all aspects such as feature engineering, the approaches to training the ML models and the obtained results allows us to start working on improving the AI-controlled computer players in Tactical Troops: Anthracite Shift. On the one hand, we may aim at making them less predictable. In this case, the trained prediction models will serve as baselines that the bots will be trying to "fool". On the other hand, we can use the prediction models within the bots' algorithms to incorporate their opponent's modelling. This way, computer players will have a better assessment of the current game situation. The diversity of both approaches leads us to a conclusion that win-prediction task has many practical applications. As motivated in the introduction, many business problems can be *gamified*, i.e., represented in a game-like framework. In such cases, we only need to rephrase the concept of winning (in terms of some goal to achieve) and the use of win-prediction models can be similar to a game scenario.

The presented post-competition data analysis and visualizations are examples of functionalities that we are permanently including in the KnowledgePit platform. We believe that they will bring benefits to future competition organizes, sponsors, and also contestants by providing better insights regarding considered data sets and solutions. In particular, the analysis of solution similarities with regard to errors, and the ensemble composition can help in developing the most practical solution to any competition task.

References

[1] M. Bowling, J. Fürnkranz, T. Graepel, and R. Musick, "Machine Learning and Games," *Machine learning*, vol. 63, no. 3, pp. 211–215, 2006.

[2] M. Świechowski and D. Ślęzak, "Grail: A Framework for Adaptive and Believable AI in Video Games," in *2018 IEEE/WIC/ACM International Conference on Web Intelligence (WI)*. IEEE, 2018, pp. 762–765.

[3] A. Janusz, D. Ślęzak, S. Stawicki, and K. Stencel, "SENSEI: An Intelligent Advisory System for the eSport Community and Casual Players," in *2018 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2018, Santiago, Chile, December 3-6, 2018*, 2018, pp. 754–757. [Online]. Available: https://doi.org/10.1109/WI.2018.00010

[4] P. Makanawala, J. Godara, E. Goldwasser, and H. Le, "Applying Gamification in Customer Service Application to Improve Agents' Efficiency and Satisfaction," in *International Conference of Design, User Experience, and Usability*. Springer, 2013, pp. 548–557.

[5] M. Świechowski, K. Godlewski, B. Sawicki, and J. Mańdziuk, "Monte Carlo Tree Search: A Review of Recent Modifications and Applications," *arXiv preprint no 2103.04931*, 2021, submitted to Springer-Nature AI Reviews.

[6] R. Lorentz, "Early Playout Termination in MCTS," in *Advances in Computer Games*. Springer, 2015, pp. 12–19.

[7] V. Volz, G. Rudolph, and B. Naujoks, "Demonstrating the Feasibility of Automatic Game Balancing," in *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, 2016, pp. 269–276.

[8] M. Świechowski, R. Tyl, and D. Lewiński, "Combining Utility AI and MCTS Towards Creating Intelligent Agents in Video Games, with the Use Case of Tactical Troops: Anthracite Shift," in *2021 IEEE Symposium on Computational Intelligence for Human-like Intelligence (SSCI CIHLI)*. IEEE, 2021, (accepted).

[9] S. Gelly and D. Silver, "Monte-Carlo Tree Search and Rapid Action Value Estimation in Computer Go," *Artificial Intelligence*, vol. 175, no. 11, pp. 1856–1875, 2011.

[10] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, T. Lillicrap, K. Simonyan, and D. Hassabis, "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018. [Online]. Available: https://science.sciencemag.org/content/362/6419/1140

[11] B. Arneson, R. B. Hayward, and P. Henderson, "Monte Carlo Tree Search in Hex," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, no. 4, pp. 251–258, 2010.

[12] M. Swiechowski and D. Slęzak, "Granular Games in Real-Time Environment," in *2018 IEEE International Conference on Data Mining Workshops (ICDMW)*, 2018, pp. 462–469.

[13] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD'16. New York, NY, USA: ACM, 2016, pp. 785–794. [Online]. Available: http://doi.acm.org/10.1145/2939672.2939785

[14] OpenAI, :, C. Berner, G. Brockman, B. Chan, V. Cheung, P. Dębiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, R. Józefowicz, S. Gray, C. Olsson, J. Pachocki, M. Petrov, H. P. d. O. Pinto, J. Raiman, T. Salimans, J. Schlatter, J. Schneider, S. Sidor, I. Sutskever, J. Tang, F. Wolski, and S. Zhang, "Dota 2 with large scale deep reinforcement learning," 2019.

[15] H. Xiao, Y. Liu, D. Du, and Z. Lu, "WP-GBDT: An Approach for Winner Prediction using Gradient Boosting Decision Tree," in *2021 IEEE International Conference on Big Data, BigData 2021, USA, December 15-18, 2021*, 2021.

[16] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree," in *31st International Conference on Neural Information Processing Systems*, ser. NIPS'17. USA: Curran Associates Inc., 2017, pp. 3149–3157. [Online]. Available: http://dl.acm.org/citation.cfm?id=3294996.3295074

[17] Q. H. Vu, D. Ruta, L. Cen, and M. Liu, "A combination of general and specific models to predict victories in video games," in *2021 IEEE International Conference on Big Data, BigData 2021, USA, December 15-18, 2021*, 2021.

[18] K.-S. Tseng, "Predicting Victories in Video Games: Using Single XGBoost with Feature Engineering," in *2021 IEEE International Conference on Big Data, BigData 2021, USA, December 15-18, 2021*, 2021.

[19] R. Roelofs, S. Fridovich-Keil, J. Miller, V. Shankar, M. Hardt, B. Recht, and L. Schmidt, *A Meta-Analysis of Overfitting in Machine Learning*, 2019.

[20] A. Janusz, "Combining Multiple Predictive Models Using Genetic Algorithms," *Intelligent Data Analysis*, vol. 16, no. 5, pp. 763–776, 2012. [Online]. Available: https://doi.org/10.3233/IDA-2012-0550