# Personalized Game Difficulty Prediction Using Factorization Machines

Jeppe Theiss Kristensen
jetk@itu.dk
IT University of Copenhagen
Digital Design
Copenhagen, Denmark

Christian Guckelsberger
christian.guckelsberger@aalto.fi
Aalto University
Department of Computer Science
Espoo, Finland

Paolo Burelli
pabu@itu.dk
IT University of Copenhagen
Digital Design
Copenhagen, Denmark

Perttu Hämäläinen
perttu.hamalainen@aalto.fi
Aalto University
Espoo, Finland

## ABSTRACT

The accurate and personalized estimation of task difficulty provides many opportunities for optimizing user experience. However, user diversity makes such difficulty estimation hard, in that empirical measurements from some user sample do not necessarily generalize to others.

In this paper, we contribute a new approach for personalized difficulty estimation of game levels, borrowing methods from content recommendation. Using factorization machines (FM) on a large dataset from a commercial puzzle game, we are able to predict difficulty as the number of attempts a player requires to pass future game levels, based on observed attempt counts from earlier levels and levels played by others. In addition to performance and scalability, FMs offer the benefit that the learned latent variable model can be used to study the characteristics of both players and game levels that contribute to difficulty. We compare the approach to a simple non-personalized baseline and a personalized prediction using Random Forests. Our results suggest that FMs are a promising tool enabling game designers to both optimize player experience and learn more about their players and the game.

## CCS CONCEPTS

• **Human-centered computing** → **User models**; *Human computer interaction (HCI).*

## KEYWORDS

Factorization Machines, games, player modelling

**ACM Reference Format:**
Jeppe Theiss Kristensen, Christian Guckelsberger, Paolo Burelli, and Perttu Hämäläinen. 2022. Personalized Game Difficulty Prediction Using Factorization Machines. In *The 35th Annual ACM Symposium on User Interface Software and Technology (UIST '22), October 29-November 2, 2022, Bend, OR, USA.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3526113.3545624

## 1 INTRODUCTION

Understanding and estimating task difficulty is a fundamental problem in Human-Computer Interaction (HCI). While good user interface design aims to minimize the difficulty of completing tasks and achieving goals, we might also be interested in introducing challenges of just the right difficulty – neither too easy nor too hard – to e.g. support learning [54] or create enjoyable video games [8, 49]. Estimating how difficult a challenge is for the target user is hard due to the diversity of factors affecting difficulty. On a macro level, the difficulty of making decisions can be boiled down to skill, the available time, and the inherent difficulty of the decision [2]. However, only the available time is straightforward to measure in general, and in HCI, executing decisions can pose additional perceptual-motor difficulties.

This paper investigates difficulty in the particular setting of casual mobile puzzle games where the player progresses in the game by completing discrete challenges, or levels. In this context, a common operationalization of difficulty is the chance of the player completing the level, or pass rate. This is typically calculated as the count of successful attempts divided by the total attempts on the level. The inverse represents the average number of attempts per complete. Averaged over all players, this is a useful difficulty metric for game designers since it gives a tangible measure of how much time a player spends on a level before proceeding, which can help identify where the player might feel stuck [14] as a potential cause for churn. However, a significant drawback of this measure is that it does not account for individual differences in skill. Simply looking at the aggregate population, therefore, runs the risk of alienating anybody but the average player who, in many cases, is neither representative nor the most important prediction target. A more useful approach for game designers should thus take both individual player and level information into account.

*UIST '22, October 29-November 2, 2022, Bend, OR, USA*

ACM ISBN 978-1-4503-9320-1/22/10…$15.00
https://doi.org/10.1145/3526113.3545624

This challenge of accounting for individual difficulties has been approached in multiple ways. A common practice is to utilize *dynamic difficulty adjustment* (DDA) in which the predictions are used to automatically adjust game parameters (e.g. number of enemies) and tailor the game difficulty to maximize aspects such as retention or monetization of individual players. However, this kind of fine-grained control over the levels is not always possible due to technical aspects (difficult to implement, uncertainty about parameters' effect on difficulty, etc.) or level and game design choices (levels requiring a specific strategy or visuals, difficulty curve must follow a certain pattern, etc.). Moreover, fully automated difficulty adjustment might not be attractive for game designers if they want to retain some control over the player experience. An alternative approach that can be of practical use should therefore also capture and explain player-level interactions and generate knowledge for the game designers that allows them to be more proactive.

The main objective of this paper is to showcase such a framework that game designers can use to both understand the interaction between players and levels and to estimate level difficulty for individual players. Rather than updating the estimates between game rounds, as is common in many DDA approaches, the goal of this framework is to leverage daily play session data to inform the offline work of level designers and help them understand the player base. To achieve this, we use *factorization machines* (FMs) which are especially known from recommendation systems [20, 39, 40, 42]. FMs allow predicting user-content interactions by estimating latent variables that describe each user and each piece of content.

To better understand how FMs can be applied for difficulty estimation, we investigate the following research questions:

**RQ1:** How do FMs compare to other difficulty prediction methods?
**RQ2:** How many observations of a player are necessary before it is possible to discern them from the average player?
**RQ3:** What do the FM model latent variables mean or represent?

*Contribution.* In summary, we examine FMs as a novel approach for personalized game level difficulty prediction. Using a large dataset of 700k players from the commercial puzzle game Lily's Garden, we compare FMs against both a naive non-personalized baseline and personalized predictions using Random Forests (RF). Our results support the use of FMs as a promising tool that clearly outperforms the other methods, especially if augmented with similar additional features as in the RF regression.

## 2 RELATED WORK

To contextualize our contribution, we first resolve ambiguity around the concept of difficulty, and survey related work on operationalizing and quantifying difficulty in videogames. We then survey existing player difficulty prediction models.

### 2.1 Game Difficulty

Game difficulty is a highly ambiguous concept [12], with at least two meanings [11]. Firstly, it can denote an *intrinsic attribute* of a game, characterizing a game-internal task based on its objective and the barriers that prevent potential players from achieving it. Secondly, it can describe a *relational attribute* between the game and the player, characterizing the player's experience of the task based on their individual skill and history. Often, researchers use the notion of *perceived difficulty* to convey the second, experiential meaning. To complicate things further, difficulty is often used synonymously with *challenge*. However, we can draw a subtle distinction based on the concepts' *valence* [11]: players typically consider a game task difficult, if it causes them frustration and discomfort; challenging tasks in contrast are stimulating and convey a feeling of being in control over the outcome [26]. Here, we primarily use the notion of difficulty, but without appealing to its negative valence.

Difficulty can be actively sought by players as a goal experience [6], or it can form the foundation [36] of other experiences. Famously, perceived difficulty contributes to player *enjoyment*, as investigated by Alexander et al. [1]. Another such goal experience is *flow* – a state in which a player feels engulfed in the task and loses track of time and worries [8, 10]. It results from exposing a player to difficulties that are optimal with respect to their individual skills. Self-determination theory [45, 50] posits that optimal difficulty satisfies players' intrinsic need for feelings of competence and in effect yields motivating gameplay. Amongst others, flow and intrinsic motivation contribute to player engagement [4], which, similar to enjoyment, constitutes a core game design objective.

Given the many ways through which difficulty impacts player experience, it is unsurprising that game designers and researchers have sought ways to operationalize difficulty for use in design-time quality control or runtime optimization. In the particular case of puzzle games, Pusey et al. [37] proposed several objective measures to quantify difficulty, including the number of incorrect attempts, the number of actions used, and the time taken to solve a puzzle. The average number of attempts that players spend to complete a level, or inversely the pass rate, has been used to assess difficulty in puzzle games such as Angry Birds [43] and Lily's Garden [24]. A player's experienced number of successes and failures also constitutes a major component of perceived difficulty, as empirically identified by Denisova et al. [11] in the development of a questionnaire to assess perceived difficulty in games. Similar to previous work, we operationalize an individual player's perceived difficulty as the average attempts per complete. Given that players cannot repeat levels in this study, this is equivalent to the number of attempts they require to complete a specific level.

### 2.2 Predicting Difficulty

Previous research has approached player difficulty prediction in numerous ways depending on the application and research purpose. Common application areas of difficulty prediction include DDA and automated playtesting for quality assurance. We next identify several shortcomings of existing work based on selected examples and highlight how our contribution overcomes them.

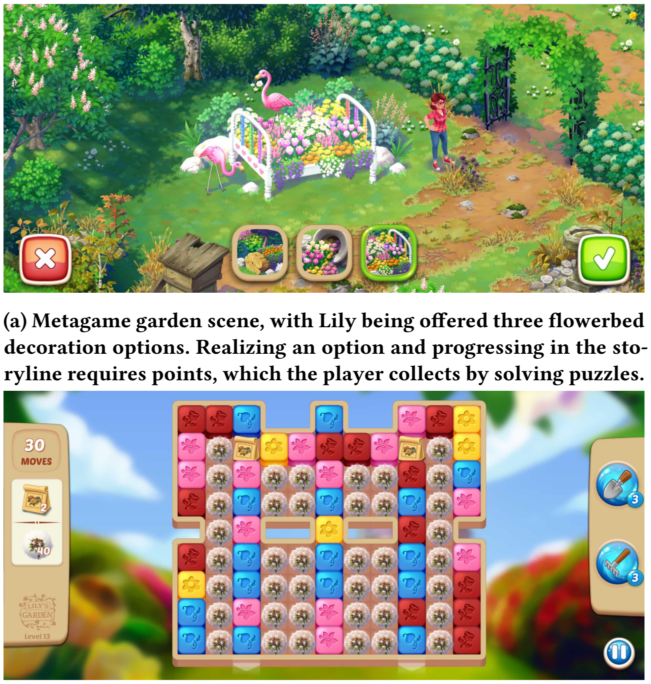
Existing approaches typically predict difficulty aggregated over a player population rather than individual experiences (e.g. [18, 22, 24, 32, 43, 51]). An example of this is the work by Gudmundsson et al. [18] where an AI game-playing agent was used to extract a preliminary estimate of the level pass rate. This estimate was then combined with level features to create a binomial regression model to predict the overall pass rate on the level. We consider the focus on aggregated predictions a shortcoming, as the perceived difficulty is affected by individual skill and experience, and an aggregate prediction is thus likely to predict the various goal experiences that perceived difficulty contributes to less accurately. In contrast to existing approaches, this paper focuses on predicting individual player difficulty, thus moving one step closer to predicting perceived difficulty as a relational attribute between one player and the game.

Existing studies that focus more on individual player difficulty prediction often suffer from practical and technical barriers: they are either operating on a very small scale (e.g. [29, 46, 47, 53]), only consider toy problems (e.g. [17]), or both (e.g. [16, 21]). Gonzalez-Duque et al., [16] for instance, use a Bayesian optimization approach to reliably estimate the time it would take a player to complete a Sudoku level given the number of pre-filled digits or a simple puzzle level given two level descriptors after having observed the player for 5 or 15 game rounds, respectively. However, the limited number of play traces (<300) in each game lead to a large variance in the results. It is uncertain how the approach would scale in a live game with several player cohorts and continuous updates to the game. In our study, we use data from more than 700,000 players over a 6 month period to demonstrate the applicability of our approach to a large-scale, complex commercial game system.

A common approach in many DDA methods is to consider the recent history of players in order to adapt the game content. However, not all methods are able to take advantage of the wealth of information that is available about the players and levels (e.g. [5, 16, 27, 55, 56]). For instance, Xue et al. [55] use a probabilistic graph that estimates the probability of winning, failing and churning based only on the player's progress and the current number of attempts; however, they do not leverage more descriptive features, such as average playtime, that have been found to be correlated with player engagement [13, 23]. In our approach, we aim to leverage a maximum range of player and level data, combined with high-cardinality data such as individual player-level interactions.

Especially in the domain of educational games, it is common to explicitly model a learning curve and introduce new elements through tutorials to build up player competencies for solving more complex tasks later [28]. The Additive Factors Model [7] is a popular method in this context for determining a player/student's chance of success on a task that requires certain skills [19]. However, many related approaches require labelling the required skills beforehand, and more specialized domain models can be hard to validate and generalize [15]. In this work, we do not assume any specific structure of player skill and knowledge and instead learn latent representations which we can interpret afterward.

As last two shortcomings, we note that, especially in live game operations, it is not desirable to have a black-box prediction method that requires expert knowledge to operate [30], as in related work that utilizes deep learning methods (e.g. [31, 33, 35]). In addition, methods that require complex implementations (e.g. [52]) are also undesirable, as any unexpected behavior may be hard to troubleshoot and make game designers lose trust in the system. Moreover, little to no knowledge is generated that enables game designers to make informed decisions about their games. In this work, we demonstrate that factorization machines (FMs) afford ease of use, do not require any special input other than data about the player and the number of attempts they spent on a given level, and afford straightforward interpretation that game designers may be able to use for other tasks such as personalized offers or churn prediction.



**(a) Metagame garden scene, with Lily being offered three flowerbed decoration options. Realizing an option and progressing in the storyline requires points, which the player collects by solving puzzles.**

**(b) Example of a puzzle level. The goal is to collect certain board pieces, shown on the left side, within a limited number of moves.**

**Figure 1: Lily's Garden: interplay of meta and puzzle game.**

To determine feasible approaches for difficulty prediction in games, it is worth looking beyond games. A similar problem to difficulty prediction is student grade prediction [38, 48]. Sweeney et al. [48] use a number of methods to predict the grades for new students in future courses, ranging from average course grades, over Random Forest (RF) regressors, to FMs. They found that FMs performed the best when not including any other information than the student-course interaction, while with additional information, FMs and Random Forest regressors performed similarly well. We are inspired by this work and the ability of FMs to capture high-cardinality data like user-item-context interactions. Consequently, we adopt a similar strategy and compare three different approaches to modeling player difficulty, including RF and FM.

## 3 CASE STUDY: LILY'S GARDEN

We study the use of FMs for predicting individual player difficulty in the commercial game Lily's Garden by Tactile Games. Released in early 2019, Lily's Garden is a casual mobile puzzle game with more than 6,000 levels and one million daily active users worldwide.

The gameplay has two main components (Fig. 1). In a narrative-driven meta game, the player is confronted with an abandoned garden in which they can unlock new areas, make decorative choices and progress in the story by spending points. These points can be acquired by solving successive puzzle game levels unlocked at specific times in the storyline. This study focuses on predicting individual player difficulty for these puzzle levels.

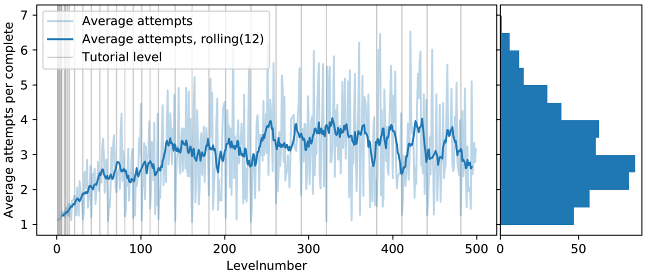


**Figure 2: The average number of attempts per level completion for the first 500 levels. The difficulty trend is illustrated as a moving mean with a centered window of size 12 – twice the length of typical designed level sequences.**

To complete a puzzle, the player must collect specific goal pieces on the board within a given move limit. The core gameplay consists of tapping on board piece clusters to clear them, destroy adjacent pieces, and hereby collect the goal pieces. By tapping on clusters with more than five, eight or ten pieces, the player can create power pieces capable of clearing large parts of the board. In some levels, forging such power pieces is strictly necessary to succeed, and more advanced strategies involve their combination for enhanced effects.

The game implements a free-to-play model. The player can use in-game and real currency to buy help in the form of power pieces and other boosters. Additionally, there are certain game events that provide such boosters for free. Moreover, players can purchase an additional five moves for the current attempt if they fail to complete all the goals within the initial move limit. While purchases are the main monetization avenues of the game, the designers ensure that every level can be completed without bought assistance.

For our model, we adopt the existing operationalization of level difficulty as the average number of attempts that players require to complete a level. An analysis of player data from Lily's Garden (Sec. 5 provides an overview of this data) shows the average number of attempts over the whole level range (Figure 2). The first few levels (< 10) contain multiple tutorial levels where the gameplay is streamlined and players are restricted to certain moves. After these levels with almost guaranteed wins, the difficulty slowly ramps up – a common design pattern to engage players early on [28].

To understand what level design aspects can affect (intrinsic) difficulty, we interviewed the team of level designers of Lily's Garden and identified a number of candidate level features that could be relevant for predicting difficulty. This includes quantifiable features such as the move limit (higher limits lead to easier levels [25]), the number of goals, and the entropy of the pieces' color distribution (the closer to uniform, the harder the level due to power pieces being harder to create). Other features, such as the level layout, board piece complexity, or reliance on power pieces, are harder to quantify but will still affect the difficulty in non-trivial ways. Defining descriptors that can fully encapsulate these intricacies is therefore rather challenging and can lead to less expressive and accurate difficulty prediction models.

In addition to the strong fluctuations in the average number of attempts over the level range, a more detailed analysis also highlights large differences between individual completion rates both with respect to the players and the levels (Fig. 3). In early, less (intrinsically) difficult levels with fewer than 1.5 attempts per complete on average, most people (>90%) require 1 attempt, with the remaining players requiring a little more. For later levels with greater (intrinsic) difficulty, we find a large variance in attempts and a long tail distribution extending to more than 30 attempts per complete. These individual differences, paired with strong fluctuations in average difficulty, make Lily's Garden an ideal, challenging candidate for studying personalized player difficulty prediction.

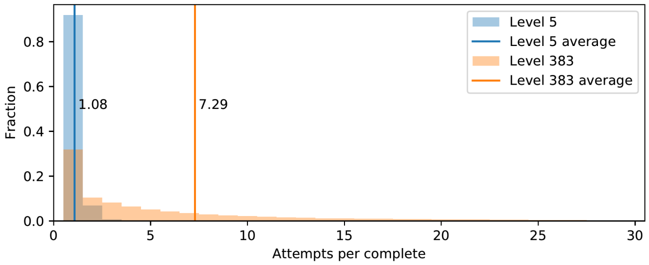


**Figure 3: Comparison of player attempt distributions for a tutorial (level 5) and a hard level (level 383).**

## 4 METHODS

In accordance with existing work on puzzle games (Sec. 2.1), we operationalize difficulty by the number of attempts a player will spend on the level. We consequently frame our individual player difficulty prediction task as a regression problem, where the target is to estimate the number of attempts a specific player will spend on a specific level. For this purpose, we compare four different methods:

- Naive baseline (**NB**): Average attempts by other players.
- Random Forest regression (**RF**): Ensemble prediction from multiple decision trees that utilize aggregated player behavior data over the observed levels.
- Factorization Machines (**FM**): A general regression model that uses a feature embedding to describe interactions between variables (e.g. user-item interactions).
- Factorization Machines with Relational Data (**FM+feat**): As **FM**, but also includes the descriptive variables used in the RF method (e.g. user-item-feature interactions).

To answer our first research question (Sec. 1), we compare these models based on their prediction error. We moreover analyze how this error changes based on the number of levels that we observed the players for. In other words, we identify the number of required observations to push the error below a certain threshold and thus answer our second research question.

### 4.1 Naive Baseline

Given that much related work focuses on predicting difficulty for a player population rather than individuals, we chose the player-average number of attempts per level complete as our naive prediction baseline. We calculate this non-personalized prediction on the players' data from our training set as illustrated in Fig. 2 using a linear regression model,

$$\hat{y} = w_0 + \sum_{\ell=1}^{L} w_\ell x_\ell, \tag{1}$$

where $w_0 = 0$, $w_\ell$ is the attempts on level $\ell$ averaged over all other players, L is the total number of levels, and $x_\ell \in \{0, 1\}$.

This non-personalized baseline is what game designers currently use in practice for estimating level difficulty. Hence, any improvements over this baseline can directly inform game designers of the compared methods' benefits.

## 4.2 Random Forest Regression

As mentioned in Sec. 2.2, a Random Forest (RF) regression model has previously been shown to deliver comparable performance to FMs [48] in a related task. Consequently, we train an RF regressor on player and level features in our comparison.

RF is an ensemble method based on multiple random trees, i.e., decision trees $h(\mathbf{x}; \theta_t), t = 1, ..., T; \theta_t$ with *i.d.d.* random vectors, where the nodes of each tree are split using a random set of features and subsets of the data. For regression problems, the split is decided based on which feature leads to the largest decrease in the absolute or squared error. The final prediction then combines the predictions from the trees into an average prediction, $y = \hat{h}(\mathbf{x})$. This ensemble approach enables modeling more complex non-linear behavior and is less likely to overfit compared to a single random tree.

We use the Random Forest regressor implementation from the *scikit-learn* library (version 1.0.2) [34] with the following hyperparameters and settings: `n_estimators=150`, `max_depth=None`, `min_samples_split=2`, `min_weight_fraction=0.0`, `max_features="auto" [=n_features]`, `max_leaf_nodes=None`, `min_impurity_decrease=0.0`. Due to the size of our data, we employ an incremental training method[1].

## 4.3 Factorization Machine

Factorization machines (FMs) are a class of factorization models that can be used as a general predictor for classification, regression, and ranking tasks [39]. They are similar to linear regression models but instead model second-order terms as an interaction between variables using a feature embedding:

$$\hat{y} = w_0 + \sum_{i=1}^{n} w_i x_i + \sum_{i=1}^{n} \sum_{j=i+1}^{n} \langle \mathbf{v_i}, \mathbf{v_j} \rangle x_i x_j, \tag{2}$$

where $w_0$ is the 0th order term or global bias, $w_i$ is the first-order term and describes the bias of the $i$'th variable, and $\mathbf{v_i}$ is a second-order feature embedding vector of the $i$'th variable. $\langle \mathbf{v_i}, \mathbf{v_j} \rangle$ describes the interaction between two variables as the dot product:

$$\langle \mathbf{v_i}, \mathbf{v_j} \rangle = \sum_{f=1}^{k} v_{i,f} \cdot v_{j,f},$$

where $k$ is the number of latent factors and a hyperparameter that must be chosen.

This learned embedding is what enables modeling unseen interactions in the data, which also makes FMs widely used for recommendation systems, including games recommendations [3, 9], where user-item interactions are typically very sparse.

The input data is not restricted to the rating-user-item format. It is possible to use other contextual information [41], including

- One-hot encoding of previous user interactions on other items.
- User and item descriptors (both numeric and categorical, e.g. age or labels).
- Implicit feedback data.

Adding additional features typically increases the data set complexity by $n \times f$, where $n$ is the dataset length and $f$ is the number of additional features. However, using FMs with a *relational data* block structure to make use of repeated patterns, as suggested by Rendle et al. [41], can greatly reduce the computational complexity and make the method scale to very large datasets. Additionally, Rendle et al. found that FMs with such relational data showed a consistent performance increase over FMs without relational data.

We use the original implementation of LibFM by Steffen Rendle [40]. We train each model for 1000 iterations using Markov Chain Monte-Carlo (MCMC) with an initial standard deviation of 1 to sample $v_i$ for FM models and 0.1 for FM+feat models. Based on brief experiments and the nature of the attempt distribution data, we do not include a global bias term $w_0$.

## 5 DATA

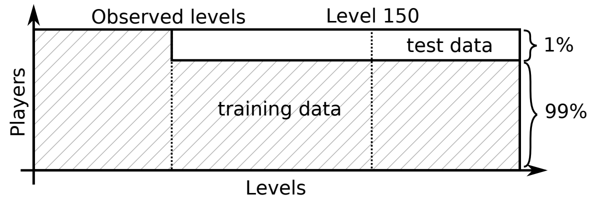


**Figure 4: Train/test data split illustration. All data of 99% of players is used for training. Additionally, the training utilizes initial observations ("Observed levels") from the 1% of players who constitute the test set.**

The data used in this study was collected from 2021-06-01 to 2021-11-30 from the game Lily's Garden and consists of 759,382 players who have all, in this period, played the game beginning with the first level and at least up to level 200. In free-to-play mobile games, it is common to have a large churn rate at the beginning of the game from players that do not interact meaningfully with the game, so this condition ensures both, that the whole history of each player is complete, and that the included players share the same minimum engagement level. Due to the long tail of the attempts distribution (Fig. 3), for numerical stability, we truncate attempts with more than 30 attempts to 30, which affects 0.34% of the data.

We split the data to match a realistic use case: We select 1% of the players to represent "new" players to test the methods on, and the remaining 99% of the players are considered "old" players who have started playing earlier and have already progressed far in the game. We use this old player data for training all models.

[1] https://github.com/garethjns/IncrementalTrees

| Type of feature | Name | Description |
|---|---|---|
| Player features | Attempts | Number of attempts on levels |
| | Moves used | Number of moves used relative to the move limit when completing the level |
| | Pre-game boosters | Boosters that can be used before starting the level |
| | In-game boosters | Boosters that can be used while playing the level |
| | Powerpieces, total | Number of power pieces created while playing the level |
| | Powerpieces, combos | Number of power piece combinations created while playing the level |
| | Rockets, solo | Number of rockets created and used on their own |
| | Rocket-bomb combo | Number of rocket-bomb combos created and used |
| | Rocket-magic combo | Number of rocket-magic combos created and used |
| | Bomb-magic combo | Number of bomb-magic combos created and used |
| Level attributes | Attempts | Average number of attempts on level by players in training set |
| | Color entropy | Entropy of color spawning weights; $S = -\sum_i p_i \log p_i$ |
| | Colors | Number of unique colors in the level |
| | SpreadingBlocker, cg | Levels with a spreading blocker as collect goal (cg) |
| | LayerCake, cg | Levels with a specific blocker with 3 hitpoints as a collect goal |
| | ConsecutiveBlocker, cg | Levels with a blocker that requires two attacks in a row as a collect goal |
| | MegaMultiColorBlocker | Levels with a large blocker that requires matching multiple colors to remove |
| | Teleport | Levels with a teleport mechanic that transports pieces around the board |

**Table 1: Features investigated for RF and FM+feat. The player features are aggregated means on the first $n$ observed levels.**

Additionally, as illustrated in Fig. 4, FM training also utilizes some initial observations of the new players. This corresponds to the model being periodically updated to improve its predictions as new players progress through the game and more observations become available. It is also necessary for FM: without observation of the given player during training, the model does not learn the bias and embedding of this player (cold start problem).

In our performance evaluations, we report the results with different numbers of initial observations. The results are always computed from levels after 150 to maintain a consistent test set even when the number of initial observations changes.

The RF and FM+feat methods require feature vectors that describe the players and levels. These features were selected based on domain knowledge from level designers and are shown in Table 1. RF methods do not deal well with high-cardinality data, so it is not possible to one-hot encode players and levels in this case. Instead, the player features for all players are aggregated and averaged across the first $n$ observed levels. This means that any prediction of a player depends on their early performance in the game and does not take recent observations into account. Otherwise, players are not comparable through their features due to each player being at different stages of the game with different types of levels, difficulties, required strategies, etc. The level attributes are static and do not change depending on the number of observations.

## 6 RESULTS

In the following, we first describe the prediction task to put the baseline prediction and error metric in context. We then identify how many observations are necessary to beat the baseline to answer our first and second research questions. Lastly, in order to answer our third research question, we provide an interpretation of the model parameters and identify key game levels for understanding the model and thus providing valuable insights for the level designers.

### 6.1 Baseline Prediction and Error

Both the RF and FM models have been optimized using root mean square error (RMSE). However, the underlying distribution of attempts, as shown in Fig. 3, follows a geometric-like distribution, where the most common value is 1, and especially hard levels exhibit a long tail that drives the average attempts up. This long tail on hard levels can yield a large RMSE on said levels, leading to the hard levels having a large effect on the model optimization. To provide a more complete picture of the models' performance and support model comparisons on easier levels, we also report the mean absolute error (MAE) next to RMSE.

Crucially, we cannot expect this baseline nor any of our methods to yield a close-to-zero prediction error. This is because each outcome of a level playing attempt is also governed by aleatoric uncertainty, which the model cannot account for. Each prediction captures the expected value for a given player-level combination.

Calculating the baseline error by aggregating all data points, we find the prediction errors across the whole level range to be $\text{RMSE}_{\text{all}} = 3.86$ and $\text{MAE}_{\text{all}} = 2.33$. The errors after level 150 are $\text{RMSE}_{\ell>150} = 4.10$ and $\text{MAE}_{\ell>150} = 2.53$. The larger error on $\ell > 150$ is due to these levels being generally (intrinsically) more difficult (Fig. 2) and thus having a larger attempt variance.

### 6.2 Effect of Observed Level Count

Before being able to differentiate between players meaningfully, we require sufficiently many discernible observations of their gameplay. The first 10 levels introduce the core gameplay, and new mechanics are then introduced in every tenth tutorial level (21, 31, 41, ... see Fig. 2). We, therefore, compare the methods when trained at 6 points of

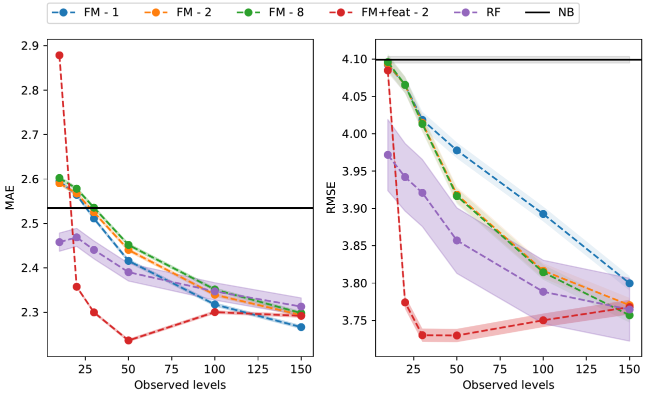


**Figure 5: MAE and RMSE on test levels $\ell > 150$ for different observed level counts. The shaded area shows the 95% confidence interval around the means.**

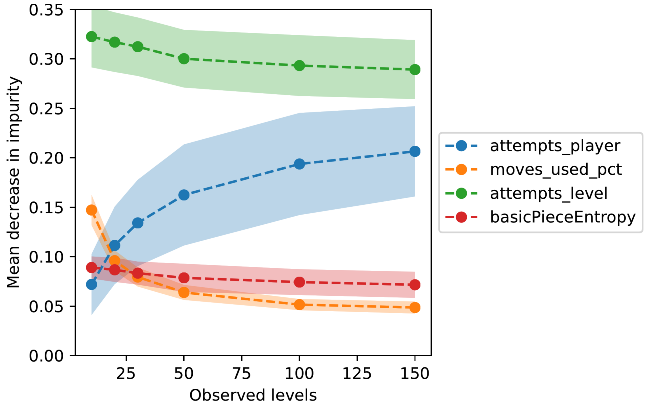


**Figure 6: Mean feature importances and one standard deviation for the Random Forest (RF) regressor. Only features with a relative significance above 0.05 are shown.**

a player’s progress: at 10, 20, 30, 50, 100 and 150 levels. To avoid information leaking between training and test sets, we evaluate the predictions on the test users after level 150.

We tested our FMs with 1, 2, 4, 8, 16, and 32 factors, but we leave out the 4, 16, and 32-factor models in the presented results for clarity of visualization since they do not alter or further inform our conclusions. The MAE and RMSE of all tested models are shown in Fig. 5, along with the 95% confidence intervals. The FM models without additional features (i.e., excluding FM+feat-2) all have similar performance, with the 1-factor model performing better in terms of MAE and the other FM models performing better in terms of RMSE. This suggests that using a single latent factor is not enough to capture the large variance in high-difficulty levels (see Fig. 3), but it makes the model less likely to overfit and perform worse on easier levels. The RMSE plot shows that these FM models are on par with the baseline prediction for as little as 10 observed levels, and, as more levels are observed, the prediction further improves over the baseline.

The predictions can be further improved while requiring fewer observations by including additional features as described in Table 1. This holds for both the RF and FM+feat models. The RM+feat-2 model shows superior performance, but its error increases after 50 observed levels. We consider this an effect of overfitting to the additional data since the training error for all the FM+feat models appears to be around $\text{RMSE}_{\text{train}} = 3.2$.

This highlights the importance of feature engineering, and while more sophisticated features could be utilized, the basic FMs presented here are game-agnostic and scale easily, providing a feasible method for game designers to employ. Our results also show that, even when additional information is available and included as features in the RF model, FMs are still able to extract more relevant information from the player-level-feature interactions.

To explore why early predictions can be improved by including additional data comprising more fine-grained descriptions of players, we analyze the feature importances from the RF model. Fig. 6 shows how the four most important features change depending on the level observation count. We find that the model utilizes additional information other than the number of attempts: early on, more fine-grained behavior data such as the average number of moves is more important compared to later on, where the average number of player attempts becomes increasingly more important. However, with more observed levels, all models reach a similar degree of performance. While the FM+feat-2 model performance appears to stagnate, the FM approaches reach similar, if not better, performance. This stagnation may be caused by the feature aggregation, where the possible level of detail and stand-out behaviors are more washed out with an increasing number of observations. Other features, such as power piece and booster usage, all have a relative importance of around 0.05, i.e., they are not uninformative. However, they are strongly correlated, which may reduce their joint importance, as mentioned in Sec. 4.2.

We also analyzed how the accuracy of predictions varies depending on the specific level they are computed for. To this end, we plotted the MAE of the FM-2 model relative to the naive baseline, as shown in Fig. 7. We omit a similar plot for the other models as they exhibited similar behavior. We find that the predictions immediately after the last observed level are more accurate, with performance deteriorating compared to the baseline on later levels.

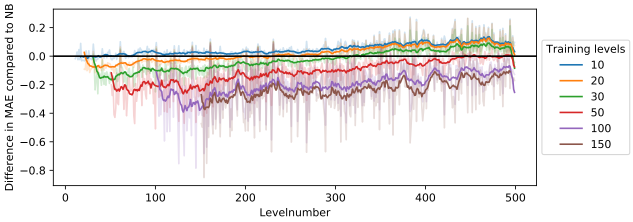


**Figure 7: MAE difference between the prediction and baseline for the FM-2 model on the test user predictions for different observed level counts. The lines show the rolling average over 12 levels. Similar behavior was observed for the other models but is not shown for clarity.**

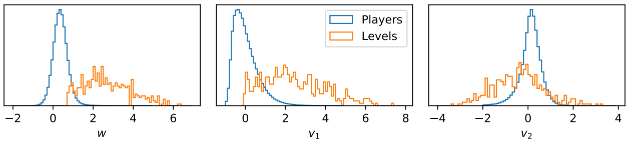


**Figure 8: Histogram of the FM model parameters (100 observed levels, 2 latent factors).**

We conclude that, firstly, the FM prediction accuracy drops at later levels, and, secondly, more observed levels lead to a slower decline in predictive performance, meaning a higher observed level horizon allows a model to be used farther into the future.

The results presented in this section allow us to answer the first two research questions: Firstly, the basic FM model without additional information fares worse than the RF model with additional data until after 100 observed levels, where they converge to similar performance. However, the FM with the same additional data as the RF outperforms all other models after 20 observed levels. Secondly, the number of observations that are necessary to discern a player from the average players depends on how detailed the available information is: with only the aggregate number of attempts, the FM method requires between 10 and 30, while more fine-grained data can enable predictions after 10 observations for all approaches.

## 6.3 Interpreting FM Model Parameters

To answer the third research question, we investigate the found FM model parameters in detail. As already mentioned, the FM models with two or more factors seem to capture other aspects than the 1-factor model, and the FM-2 method yields comparable performance as the RF method after 100 observed levels. We thus limit the analysis to the 2-factor model with 100 observed levels.

FMs use two main parameters (Sec. 4): the variable biases, $w$, and the latent factors that describe second-order interactions, $\mathbf{v}$ (ignoring subscripts for specific users/items/attributes). As a first step, we consider histograms of the model parameters in Fig. 8 and separate the distributions by player and level parameters. We find that the distributions are distinctly different between players and levels for all three parameters. We consequently differentiate between levels and players in the following examination of each parameter before discussing their relationship to the player-level interactions.

**Variable bias, w:** For the levels, this variable is strongly correlated ($\rho_{\text{Spearman}} = 0.99$) with the average number of attempts on that level (Fig. 9). However, the $w$ parameter for the levels is not sufficient to match the underlying distribution of attempts, which is not surprising since that would correspond to the baseline model. As for the players, $w$ does not correlate strongly with any player-related metrics. This suggests that the main first-order term comes from the levels, supporting an interpretation of $w$ as a baseline level difficulty. The remaining variance must therefore be explained by the second-order interaction terms between player and level.

**First latent factor, $\mathbf{v}_1$:** For both the levels and players, this variable is also strongly correlated with the average number of attempts ($\rho_{\text{Spearman}} = 0.98$ and 0.70, respectively), as well as the variance of the attempts ($\rho_{\text{Spearman}} = 0.99$ and 0.73, respectively).

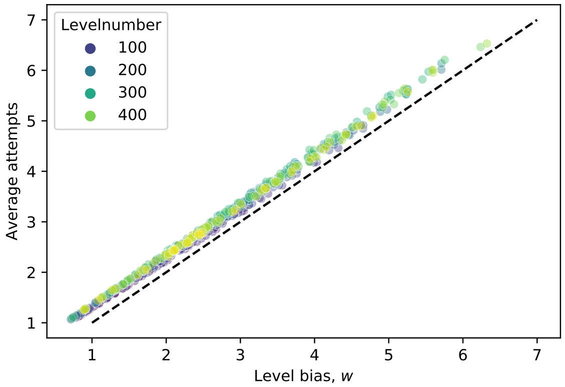


**Figure 9: The average attempts on the level over the level bias, $w$. The black dashed line is the diagonal $w$ = attempts.**

We note that for the levels, $v_1$ is strictly positive, with the exception of 5 tutorial levels which are only slightly negative. This suggests that $v_1$ captures a variance effect for each level, where the sign of the interaction depends solely on the player. The amplitude then reflects the consistency of the level/player: a value close to 0 for the level signals little variation between the players (it may only require one simple strategy to win, e.g. in tutorials), while a large $v_1$ suggests a large variation (e.g. it may require multiple strategies, or come with high aleatoric uncertainty). For the player, $v_1$ is more reflective of their skill level: for $v_1 < 0$, we expect a player to use consistently fewer attempts than their peers, especially on harder levels where skill and strategy matter more. Conversely, $v_1 > 0$ indicates that the player struggles to employ winning strategies.

To support this interpretation, we investigated levels at the extreme values of $v_1$. Since $v_1$ is strongly correlated with the average number of attempts to complete a level, which in turn is strongly correlated with the attempt variation on said level, we show the attempt variation normalized by the average attempts as a function of $v_1$ in Fig. 10. At small values of $v_1$, we find the tutorial levels which generally have a high move limit and do not require any special strategies. Conversely, at large $v_1$ are the levels with the greatest difficulty and variance compared to the mean. As an illustrative example, we show level 383 in Fig. 11, which has the highest $v_1$ among all levels. There are multiple gameplay reasons why this level might have high variance and difficulty:

1. The level contains blockers, which can only be destroyed with power pieces. Less skilled players will have a harder time creating the necessary pieces.
2. The spawning weights on piece colors are almost uniform, making it harder to create power pieces.
3. There is one specific power piece combination (2 magics, requiring 2 combinations of 10 pieces) that can easily win a level. More skilled players may have learned this strategy.
4. The board itself is larger than average, allowing more possible strategies (e.g., focus on the top/bottom/left/right side).

All these aspects lead to the perceived level difficulty being highly variable and dependable on the player's skill and game knowledge, supporting the interpretation that $v_1$ expresses how

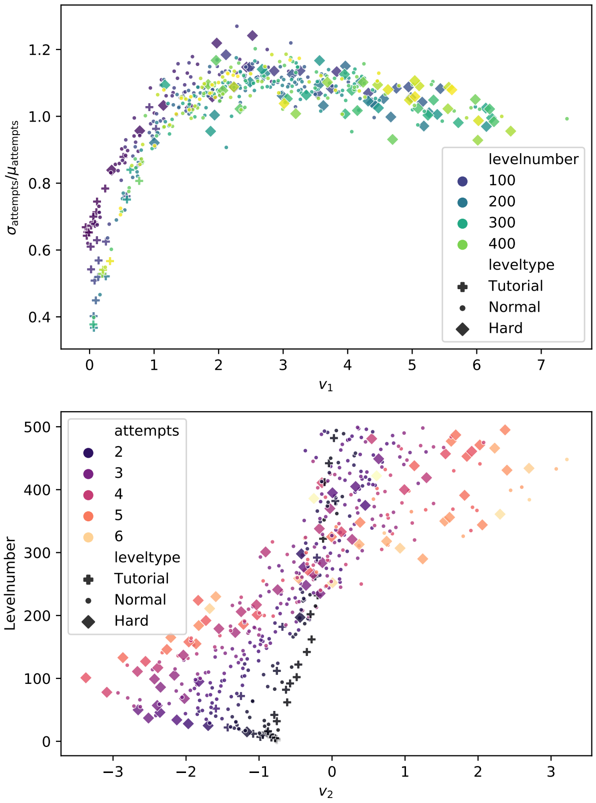
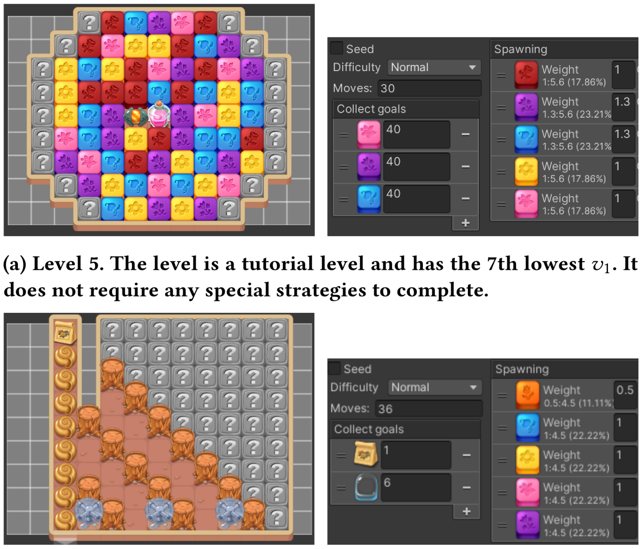


**Figure 10: Top: Normalized variance versus the latent factors, $v_1$. Bottom: The level number plotted against the second latent factor, $v_2$. Note that level designers assign the *hard* label based on their own initial estimates, so it does not always reflect whether a level requires many attempts on average.**

large an impact the player's skill can have on a given level. In contrast, level 5 in Fig. 11 has the 7th lowest $v_1$ and requires a much more simple strategy to win, affording little variation in individual player difficulty.

**Second latent factor, $v_2$:** This factor is strongly negatively correlated with the level number, as shown in Fig. 10. For levels below 250, $v_2$ tends to be negative and becomes positive beyond this threshold. There is nothing in the gameplay that changes drastically in the first 500 levels, but since the level number is linked to how many players have played the level, we interpret $v_2$ to capture aspects that are more temporal and related to the underlying data distribution: there are more observations on early levels, and different players have played later levels. It is thus likely that FM models with more factors find such spurious connections in the data.

For a deeper interpretation of $v_2$, we note that $v_2$ is inversely correlated with player features as compared to $v_1$. Since $v_1$ can be interpreted as player consistency and skill, $v_2$ appears to capture similar aspects but with the opposite sign, although it should be noted that the sign of the factors is arbitrary, as the dot product between user and content factors stays unchanged if one changes the sign of both factors. Since $v_2$ for the levels is centered around 0, the effect on the prediction depends on the player and their progress: a negative $v_2$ for a player means they spend more attempts on early levels but fewer attempts later on, while a positive $v_2$ signals a reduction in predicted attempts early on and more attempts later on. The $v_2$ factor can therefore also be related to a temporal shift in a player's playstyle compared to their peers. Crucially, this interpretation is less clear than for the first latent factor as $v_2$ appears to capture aspects more connected to the data collection rather than the player-level relation.



Seed
Difficulty: Normal
Moves: 30
Collect goals
40
40
40
+
Spawning
Weight 1:5.6 (17.86%) 1
Weight 1.3:5.6 (23.21% 1.3
Weight 1.3:5.6 (23.21% 1.3
Weight 1:5.6 (17.86%) 1
Weight 1:5.6 (17.86%) 1

**(a) Level 5. The level is a tutorial level and has the 7th lowest $v_1$. It does not require any special strategies to complete.**

Seed
Difficulty: Normal
Moves: 36
Collect goals
1
6
+
Spawning
Weight 0.5:4.5 (11.11%) 0.5
Weight 1:4.5 (22.22%) 1
Weight 1:4.5 (22.22%) 1
Weight 1:4.5 (22.22%) 1
Weight 1:4.5 (22.22%) 1

**(b) Level 383. The level is hard and has the highest $v_1$. In order to complete the level, the first level goal requires removing all the shells to the left, so the seed bag drops down to the bottom, while the second goal requires attacking either of the three bubble spawners at the bottom and then destroying six bubbles. The shells and tree stumps are both hard blockers, requiring power pieces for their removal.**

**Figure 11: Examples of a tutorial (*a*) and hard level (*b*). The left-hand side shows the level layout, and the right-hand side shows the move limit, level objectives, and color distributions. The pieces marked with '?' are assigned a random color at the start given by the weight distribution shown to the right.**

## 7 DISCUSSION

Overall, our results from this case study suggest that FM outperforms both the naive baseline and RF, especially if augmented with similar features as RF. Although RF performs better with fewer observations than non-augmented FM, the latter is game-agnostic, can be informative about the inter-dependency between players and levels, and does not require expert knowledge on relevant player and level descriptors. If such information is available, it can be utilized by FMs in addition to the player-level data. Crucially though, there remain some caveats and limitations regarding the practical use of FMs and their generalizability, which we further elaborate on below.

### 7.1 New Players and Content

One issue with FMs is the cold start problem where the model does not learn model parameters ($w$, $\mathbf{v}$) for a specific player if they were not included in the training data (e.g. if a model is trained during the night and the player starts playing the next day). This can be mitigated by including additional player information, as we tested in this study. While this increases computational complexity, the capacity to describe the player with a set of features that can be calculated immediately enables predictions without retraining the whole model.

While we did not extend our research to include unseen levels, the same cold start problem also exists in that case. We expect this to be more difficult to mitigate since information about the historical average number of attempts on a level – a very strong predictor for the individualized predictions – is unavailable, and the entropy of the color distribution (which *is* available) only explains a small percentage of the variation. More work is therefore necessary to identify relevant level features, but some immediate options could be to include AI playtest agent data since this data can be strongly correlated with the player pass rate [24, 43, 44]. We advocate FMs as a straightforward extension to the current AI playtesting approaches that will be able to capture the temporal and cohort differences, unlike previous prediction models.

### 7.2 Utility to Game Designers

A lot of work on DDA focuses on immediate adjustments to the game during play. This may be possible using an FM model and relational data; however, the strength of the FM approach is not just accurate predictions but also the modeling of further player characteristics. The latent factors were identified to be related to the players' skill and consistency over time, and this information can be relevant to many other features of the game. For instance, it could be used to cluster people together for in-game tournaments or groups so that they match in skill level and the competition is fair. It can also help level designers in proactively maintaining the level database by identifying problematic levels and bottlenecks before a new player cohort reaches this point. Ideally, in-game purchase data can also be incorporated into the process to provide estimates for both expected difficulty and monetization effectiveness and let level designers make informed decisions on necessary changes.

Before a tool like this can be implemented in a live game, though, there are a number of practical challenges that need to be considered first. Level design is very often an iterative process where minor adjustments are repeatedly performed on the level. Any change to the level will lead to a different difficulty, and the challenge is to convey this to the model. While some changes are easy to quantify, such as an alteration of the move limit or goals, others can be more tricky: even minor changes in the layout (e.g., assigning specific colors to start pieces or creating a hole in the level) can have a large impact on the difficulty, creating almost an entirely new level. We argue that FMs should be able to deal with both cases: if the change is easily quantifiable, it can be added as relational data, and if not, the level can be counted as a completely new level. While this would discard the information that two levels are very similar and increase data sparsity, FMs generally perform very well on sparse data compared to other algorithms, making them a very good candidate to use in the game industry.

### 7.3 Sensitivity to Randomness

As noted earlier, due to the stochastic nature of Lily's Garden, it is not possible to predict the exact number of attempts of a player on a level. An alternative could be to predict the probability of success per attempt, similar to Xue et al. [55]. Approaching the optimization problem from this perspective would require replacing the RMSE and MAE for evaluating the models with a measure such as the Poisson deviance, which is more appropriate for strictly positive counting data. However, for this study, we focus on RMSE and MAE, as these measures are more canonical and easier to interpret by game designers.

### 7.4 Generalizability to Other Games and Domains

Our study is based on one specific puzzle game, but we expect our approach to work for other games as well, including other types of puzzle games, different genres (e.g. platformers or first-person shooters), or even different types of games (e.g. educational games). FMs are task-agnostic and can be applied to any game where the following requirements are met:

1. The variable of interest, e.g., pass rate or time taken, can be predicted as a multiplicative (or divisive) combination of user and content latent variables such as skill and difficulty.
2. Multiple users are exposed to the same units of content such as puzzles or levels, allowing for inference of the latent variables.

Many games meet these requirements, but there are notable exceptions, such as highly/completely random games, where the player has little effect on the outcome, and procedurally generated games, where the generated content can be unique for each player. Some instances of procedural content generation might still allow modeling latent player-content relations, e.g. different players struggling with different enemies. However, this needs to be evaluated on a per-game basis.

While we expect the model parameters to capture similar aspects across games, we do not consider it feasible to directly transfer a trained model to another game unless the games are very similar in terms of both gameplay and the distribution of predictor variables. However, if the same players play multiple games, all the multi-game user-content interactions could, in principle, be combined into one big FM dataset. This is a potential direction for future research. It should be noted, though, that games can have different types of challenges, such as emotional, cognitive, and physical challenges [12]. To model all the challenge and skill facets of multiple games, one will likely need more FM factors than our default 2.

### 7.5 Generalizability to Difficulty and/or Skill Varying Over Time

Our approach does not explicitly take into account players' learning and skill improvement during play. However, Fig. 7 demonstrates that the model performs well more than 100 levels into the future. Predicting beyond such a safe horizon is typically not necessary, as the model can be retrained periodically to include new observations of each player progressing through the game and gaining in skill. For instance, in our case, it would be feasible to retrain daily, as most players consume far less than 100 levels per day (median = 7, 90th quantile = 31), and the training takes only 14 hours on an Intel Cascade Lake *c2-standard-16* Google Cloud machine.

Generalizability to dynamically varying skill and difficulty could also be improved with additional FM input features. An obvious choice is to utilize difficulty features such as a level's move limit or the amounts and types of enemies. It should also be possible to utilize player statistics such as total playtime, to allow FM to explicitly model skill acquisition over time. Future work is needed to investigate whether and how much such extra features improve prediction accuracy.

## 8 CONCLUSION

We have considered the problem of predicting personalized perceived difficulty and specifically how many attempts a player will spend on completing a level in a commercial mobile puzzle game. To this end, we compared four approaches: a simple non-personalized baseline, a Random Forest regressor, and Factorization Machines (FMs), the latter with and without game-specific features.

In terms of prediction performance, both the RF and FM methods achieved a lower MAE than the baseline after 10 and 30 observations of the players, respectively, answering our second research question about how many observations are necessary for being able to discern between players. The FM method that utilizes the same features as the RF model achieves better performance than all the other models already after 20 observations. All models had a lower RMSE than the baseline after 10 observations, indicating that any kind of personalized prediction can improve difficulty estimates.

A deeper analysis allowed us to answer our last research question about the FM parameters and their correlation with player and level characteristics. The first-order term, $w$, captures level difficulty, while the first latent factor, $v_1$, captures a player-level interaction that quantifies the player skill and level randomness: for levels, $v_1$ was strictly positive (except for 5 tutorial levels) and can be thought of as a level variance, whereas for players, both the magnitude and sign of $v_1$ indicate how sensitive and consistent a player is to the stochastic elements of a level and can thus be thought of as an expression of skill. Lastly, $v_2$ captures signals related to the underlying data distribution and temporal changes in player behavior. For levels, $v_2$ is strongly correlated with the level number. The latent factor of the player, therefore, describes whether they perform better or worse over the progress of levels compared to their peers and, in a sense, how well they learn how to play. Including more latent factors led to overfitting and was thus not analyzed in more detail.

Overall, we find that FMs have multiple advantages over other prediction approaches: they outperform other typical approaches even when only relying on game-agnostic player-level-attempt data but have the option to utilize more fine-grained data to further improve performance; the FM model parameters provide interpretable results; FMs are scalable to large amounts of data. FMs therefore show a large potential for difficulty modeling not just for research but also in an industrial context where such advantages are requirements.

## 9 FUTURE WORK

We have focused on estimating how many attempts a player will spend on completing a level, but there are a number of extensions to the model that may make it even more useful for game designers.

The model performance clearly improves with additional information, especially when only a few player observations are available. A relevant line of future investigation is therefore to identify the features that have a large impact on the model performance and optimize the feature selection to prevent overfitting. Additionally, including AI playtest agent data may also improve the predictions on levels with very few or no player-level observations.

The same player-level data could also be used in future work to predict variables other than the number of attempts. Other useful prediction targets could include churn, the probability of purchasing a continue, or the number of actions required to complete a level. Furthermore, the learned model parameters might be useful for other modeling approaches such as personalized offers, churn prediction, or sampling criteria for A/B testing new content to ensure enough player variation of the content.

## ACKNOWLEDGMENTS

Many thanks to our anonymous reviewers for their extremely valuable feedback. CG has been partly funded by the Academy of Finland Flagship program *Finnish Center for Artificial Intelligence* (FCAI). We thank Tactile Games for supporting this work, and especially the level design team for sharing their expertise and insights.

## REFERENCES

[1] Justin T. Alexander, John Sear, and Andreas Oikonomou. 2013. An investigation of the effects of game difficulty on player enjoyment. *Entertainment Computing* 4, 1 (Feb. 2013), 53–62. https://doi.org/10.1016/j.entcom.2012.09.001

[2] Ashton Anderson, Jon Kleinberg, and Sendhil Mullainathan. 2017. Assessing human error against a benchmark of perfection. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 11, 4 (2017), 1–25.

[3] Syed Muhammad Anwar, Talha Shahzad, Zunaira Sattar, Rahma Khan, and Muhammad Majid. 2017. A game recommender system using collaborative filtering (GAMBIT). In *2017 14th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. 328–332. https://doi.org/10.1109/IBCAST.2017.7868073

[4] Jeanne H Brockmyer, Christine M Fox, Kathleen A Curtiss, Evan McBroom, Kimberly M Burkhart, and Jacquelyn N Pidruzny. 2009. The Development of the Game Engagement Questionnaire: A Measure of Engagement in Video Game-Playing. *Journal of Experimental Social Psychology* 45, 4 (2009), 624–634.

[5] Sara Bunian, Alessandro Canossa, Randy Colvin, and Magy Seif El-Nasr. 2017. Modeling individual differences in game behavior using HMM. (2017).

[6] Paul Cairns. 2016. Engagement in Digital Games. In *Why Engagement Matters: Cross-Disciplinary Perspectives of User Engagement in Digital Media*, Heather O'Brien and Paul Cairns (Eds.). Springer International Publishing, Cham, 81–104. https://doi.org/10.1007/978-3-319-27446-1_4

[7] Hao Cen, Kenneth Koedinger, and Brian Junker. 2006. Learning factors analysis–a general method for cognitive model evaluation and improvement. In *International conference on intelligent tutoring systems*. Springer, 164–175.

[8] Jenova Chen. 2007. Flow in Games (and Everything Else). *Commun. ACM* 50, 4 (apr 2007), 31–34. https://doi.org/10.1145/1232743.1232769

[9] Germán Cheuque, José Guzmán, and Denis Parra. 2019. Recommender Systems for Online Video Game Platforms: the Case of STEAM. In *Companion Proceedings of The 2019 World Wide Web Conference*. ACM, San Francisco USA, 763–771. https://doi.org/10.1145/3308560.3316457

[10] Mihaly Csikszentmihalyi and Mihaly Csikzentmihaly. 1990. *Flow: The psychology of optimal experience.* Vol. 1990. Harper & Row New York.

[11] Alena Denisova, Paul Cairns, Christian Guckelsberger, and David Zendle. 2020. Measuring perceived challenge in digital games: Development & validation of

the challenge originating from recent gameplay interaction scale (CORGIS). *International Journal of Human-Computer Studies* 137 (May 2020), 102383. https://doi.org/10.1016/j.ijhcs.2019.102383

[12] Alena Denisova, Christian Guckelsberger, and David Zendle. 2017. Challenge in Digital Games: Towards Developing a Measurement Tool. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, Denver Colorado USA, 2511–2519. https://doi.org/10.1145/3027063.3053209

[13] Anders Drachen, Eric Thurston Lundquist, Yungjen Kung, Pranav Rao, Rafet Sifa, Julian Runge, and Diego Klabjan. 2016. Rapid prediction of player retention in free-to-play mobile games. (2016).

[14] Tobias Drey, Fabian Fischbach, Pascal Jansen, Julian Frommel, Michael Rietzler, and Enrico Rukzio. 2021. To Be or Not to Be Stuck, or Is It a Continuum?: A Systematic Literature Review on the Concept of Being Stuck in Games. *Proceedings of the ACM on Human-Computer Interaction* 5, CHI PLAY (Oct. 2021), 1–35. https://doi.org/10.1145/3474656

[15] Ilya Goldin, April Galyardt, et al. 2018. Most of the Time, It Works Every Time: Limitations in Refining Domain Models with Learning Curves. *Journal of Educational Data Mining* 10, 2 (2018), 55–92.

[16] Miguel Gonzalez-Duque, Rasmus Berg Palm, and Sebastian Risi. 2021. Fast Game Content Adaptation Through Bayesian-based Player Modelling. In *2021 IEEE Conference on Games (CoG)*. 01–08. https://doi.org/10.1109/CoG52621.2021.9619018

[17] Miguel González-Duque, Rasmus Berg Palm, David Ha, and Sebastian Risi. 2020. Finding Game Levels with the Right Difficulty in a Few Trials through Intelligent Trial-and-Error. In *2020 IEEE Conference on Games (CoG)*. 503–510. https://doi.org/10.1109/CoG47356.2020.9231548

[18] Stefan Freyr Gudmundsson, Philipp Eisen, Erik Poromaa, Alex Nodet, Sami Purmonen, Bartlomiej Kozakowski, Richard Meurling, and Lele Cao. 2018. Human-like playtesting with deep learning. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 1–8.

[19] Erik Harpstead and Vincent Aleven. 2015. Using Empirical Learning Curve Analysis to Inform Design in an Educational Game. In *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play* (London, United Kingdom) *(CHI PLAY '15)*. Association for Computing Machinery, New York, NY, USA, 197–207. https://doi.org/10.1145/2793107.2793128

[20] Fuxing Hong, Dongbo Huang, and Ge Chen. 2019. Interaction-aware factorization machines for recommender systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3804–3811.

[21] Martin Jennings-Teats, Gillian Smith, and Noah Wardrip-Fruin. 2010. Polymorph: dynamic difficulty adjustment through level generation. In *Proceedings of the 2010 Workshop on Procedural Content Generation in Games*. 1–4.

[22] Ildar Kamaldinov and Ilya Makarov. 2019. Deep reinforcement learning in match-3 game. In *2019 IEEE conference on games (CoG)*. IEEE, 1–4.

[23] Jeppe Theiss Kristensen and Paolo Burelli. 2019. Combining Sequential and Aggregated Data for Churn Prediction in Casual Freemium Games. In *2019 IEEE Conference on Games (CoG)*. 1–8. https://doi.org/10.1109/CIG.2019.8848106 ISSN: 2325-4289.

[24] Jeppe Theiss Kristensen, Arturo Valdivia, and Paolo Burelli. 2020. Estimating Player Completion Rate in Mobile Puzzle Games Using Reinforcement Learning. In *2020 IEEE Conference on Games (CoG)*. 636–639. https://doi.org/10.1109/CoG47356.2020.9231581

[25] Jeppe Theiss Kristensen, Arturo Valdivia, and Paolo Burelli. 2021. Statistical Modelling of Level Difficulty in Puzzle Games. In *2021 IEEE Conference on Games (CoG)*. IEEE, 1–8.

[26] R Lazzaro. 2004. Why we play games: 4 keys to more emotion. *Proc. Game Developers Conference 2004*. https://cir.nii.ac.jp/crid/1572543025651858816

[27] Jiayu Li, Hongyu Lu, Chenyang Wang, Weizhi Ma, Min Zhang, Xiangyu Zhao, Wei Qi, Yiqun Liu, and Shaoping Ma. 2021. A Difficulty-Aware Framework for Churn Prediction and Intervention in Games. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*. ACM, Virtual Event Singapore, 943–952. https://doi.org/10.1145/3447548.3467277

[28] Conor Linehan, George Bellord, Ben Kirman, Zachary H. Morford, and Bryan Roche. 2014. Learning curves: analysing pace and challenge in four successful puzzle games. In *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*. ACM, Toronto Ontario Canada, 181–190. https://doi.org/10.1145/2658537.2658695

[29] Diana Lora, Antonio A Sánchez-Ruiz, Pedro A González-Calero, and Marco A Gómez-Martín. 2016. Dynamic difficulty adjustment in tetris. In *The Twenty-Ninth International Flairs Conference*.

[30] Tobias Mahlmann, Anders Drachen, Julian Togelius, Alessandro Canossa, and Georgios N. Yannakakis. 2010. Predicting player behavior in Tomb Raider: Underworld. In *Proceedings of the 2010 IEEE Conference on Computational Intelligence and Games*. 178–185. https://doi.org/10.1109/ITW.2010.5593355 ISSN: 2325-4289.

[31] Hee-Seung Moon and Jiwon Seo. 2020. Dynamic difficulty adjustment via fast user adaptation. In *Adjunct Publication of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 13–15.

[32] Fausto Mourato, Fernando Birra, and Manuel Próspero dos Santos. 2014. Difficulty in action based challenges: success prediction, players' strategies and profiling. In *Proceedings of the 11th Conference on Advances in Computer Entertainment Technology*. 1–10.

[33] Dvir Ben Or, Michael Kolomenkin, and Gil Shabat. 2021. DL-DDA-Deep Learning based Dynamic Difficulty Adjustment with UX and Gameplay constraints. In *2021 IEEE Conference on Games (CoG)*. IEEE, 1–7.

[34] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[35] Johannes Pfau, Jan David Smeddinck, and Rainer Malaka. 2020. Enemy within: Long-term motivation effects of deep player behavior models for dynamic difficulty adjustment. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–10.

[36] Christopher Power, Paul Cairns, Alena Denisova, Themis Papaioannou, and Ruth Gultom. 2019. Lost at the edge of uncertainty: Measuring player uncertainty in digital games. *International Journal of Human–Computer Interaction* 35, 12 (2019), 1033–1045.

[37] Megan Pusey, Kok Wai Wong, and Natasha Anne Rappa. 2021. The Puzzle Challenge Analysis Tool. A Tool for Analysing the Cognitive Challenge Level of Puzzles in Video Games. *Proceedings of the ACM on Human-Computer Interaction* 5, CHI PLAY (Oct. 2021), 1–27. https://doi.org/10.1145/3474703

[38] Zhiyun Ren, Xia Ning, Andrew S. Lan, and Huzefa Rangwala. 2019. Grade Prediction with Neural Collaborative Filtering. In *2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. 1–10. https://doi.org/10.1109/DSAA.2019.00014

[39] Steffen Rendle. 2010. Factorization machines. In *2010 IEEE International conference on data mining*. IEEE, 995–1000.

[40] Steffen Rendle. 2012. Factorization Machines with libFM. *ACM Transactions on Intelligent Systems and Technology* 3, 3 (May 2012), 1–22. https://doi.org/10.1145/2168752.2168771

[41] Steffen Rendle. 2013. Scaling factorization machines to relational data. *Proceedings of the VLDB Endowment* 6, 5 (2013), 337–348.

[42] Steffen Rendle, Zeno Gantner, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2011. Fast context-aware recommendations with factorization machines. In *Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval*. 635–644.

[43] Shaghayegh Roohi, Christian Guckelsberger, Asko Relas, Henri Heiskanen, Jari Takatalo, and Perttu Hämäläinen. 2021. Predicting Game Engagement and Difficulty Using AI Players. *Proceedings of the ACM on Human-Computer Interaction* 5, CHI PLAY (Oct. 2021), 1–17. https://doi.org/10.1145/3474658 arXiv: 2107.12061.

[44] Shaghayegh Roohi, Asko Relas, Jari Takatalo, Henri Heiskanen, and Perttu Hämäläinen. 2020. Predicting game difficulty and churn without players. In *Proceedings of the Annual Symposium on Computer-Human Interaction in Play*. 585–593.

[45] Richard M. Ryan, C. Scott Rigby, and Andrew Przybylski. 2006. The Motivational Pull of Video Games: A Self-Determination Theory Approach. *Motivation and Emotion* 30, 4 (2006), 347–363.

[46] Anurag Sarkar and Seth Cooper. 2017. Level difficulty and player skill prediction in human computation games. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, Vol. 13. 228–233.

[47] Mirna Paula Silva, Victor do Nascimento Silva, and Luiz Chaimowicz. 2015. Dynamic Difficulty Adjustment through an Adaptive AI. In *2015 14th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. 173–182. https://doi.org/10.1109/SBGames.2015.16 ISSN: 2159-6662.

[48] Mack Sweeney, Huzefa Rangwala, Jaime Lester, and Aditya Johri. 2016. Next-Term Student Performance Prediction: A Recommender Systems Approach. *arXiv:1604.01840 [cs]* (Sept. 2016). https://doi.org/10.5281/zenodo.3554603 arXiv: 1604.01840.

[49] Penelope Sweetser and Peta Wyeth. 2005. GameFlow: a model for evaluating player enjoyment in games. *Computers in Entertainment (CIE)* 3, 3 (2005), 3–3.

[50] April Tyack and Elisa D Mekler. 2020. Self-Determination Theory in HCI Games Research: Current Uses and Open Questions. In *Proc. Conference on Human Factors in Computing Systems (CHI)*. ACM, 1–22.

[51] Marc Van Kreveld, Maarten Löffler, and Paul Mutser. 2015. Automated puzzle difficulty estimation. In *2015 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 415–422.

[52] Matheus Weber and Pollyana Notargiacomo. 2020. Dynamic Difficulty Adjustment in Digital Games Using Genetic Algorithms. In *2020 19th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*. IEEE, 62–70.

[53] Daniel Wheat, Martin Masek, Chiou Peng Lam, and Philip Hingston. 2016. Modeling perceived difficulty in game levels. In *Proceedings of the Australasian Computer Science Week Multiconference*. 1–8.

[54] Robert C Wilson, Amitai Shenhav, Mark Straccia, and Jonathan D Cohen. 2019. The eighty five percent rule for optimal learning. *Nature communications* 10, 1 (2019), 1–9.

[55] Su Xue, Meng Wu, John Kolen, Navid Aghdaie, and Kazi A. Zaman. 2017. Dynamic Difficulty Adjustment for Maximized Engagement in Digital Games. In *Proceedings of the 26th International Conference on World Wide Web Companion - WWW ’17 Companion*. ACM Press, Perth, Australia, 465–471. https://doi.org/10.1145/3041021.3054170

[56] Alexander Zook, Stephen Lee-Urban, Michael R. Drinkwater, and Mark O. Riedl. 2012. Skill-based Mission Generation: A Data-driven Temporal Player Modeling Approach. In *Proceedings of the The third workshop on Procedural Content Generation in Games - PCG’12*. ACM Press, Raleigh, NC, USA, 1–8. https://doi.org/10.1145/2538528.2538534