Hierarchical Transformers for Group-Aware Sequential Recommendation: Application in MOBA Games

Vladimir Araujo*
Helem Salinas
Álvaro Labarca
Andrés Villa
Denis Parra
vgaraujo@uc.cl
yhsalinas@uc.cl
aalabarca@uc.cl
afvilla@uc.cl
dparra@ing.puc.cl
Pontificia Universidad Católica de Chile
Santiago, Chile

ABSTRACT

In recent years, several recommendation systems have been introduced to improve the user experience of players in video games. In Multiplayer Online Battle Arena (MOBA) games, a popular game genre, these systems are useful for recommending items for a character during a match. Current approaches focus on recommending a fixed set of items based on a character and the other participants. However, a MOBA match is an inherently sequential process, where its past decisions define the current ones. Therefore, it would be desirable to obtain a contextual recommendation considering a specific situation and the previously consumed items. To fill this gap, in this work, we propose HT4Rec for group-aware sequential item recommendation. It consists of a contextual encoder that generates a character-item representation contextualized by the other participants involved in the game, followed by a sequential encoder that captures sequential patterns of the data to recommend the next item. In this way, HT4Rec provides a flexible and unified attention-based network structure to capture both general and long-term preferences. Our evaluations on a Dota 2 video game dataset demonstrate that HT4Rec outperforms well-known sequential recommendation methods on various evaluation metrics. Additional experiments unveil the most important parts of our model and the most relevant inputs according to the attention mechanism, which could be used to interpret the suggested items. Furthermore, we demonstrate that HT4Rec could be applied to a different scenario (movie recommendation) than MOBA games, showing better results than the baseline models.

*Also with KU Leuven.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
UMAP '22 Adjunct, July 4–7, 2022, Barcelona, Spain
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9232-7/22/07…$15.00
https://doi.org/10.1145/3511047.3537667

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → *Machine learning*.

KEYWORDS

Item Recommendation, MOBA Games, Sequential Recommendation

ACM Reference Format:
Vladimir Araujo, Helem Salinas, Álvaro Labarca, Andrés Villa, and Denis Parra. 2022. Hierarchical Transformers for Group-Aware Sequential Recommendation: Application in MOBA Games. In *Adjunct Proceedings of the 30th ACM Conference on User Modeling, Adaptation and Personalization (UMAP '22 Adjunct), July 4–7, 2022, Barcelona, Spain.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3511047.3537667

1 INTRODUCTION

Online multiplayer video games are hybrids of action games, role-playing games, and real-time strategy games. Currently, in the world of video games, the MOBA genre has become a huge representative of eSports that is projected to reach a value of 43 billion US dollars by the end of 2021[1]. One of the best-known games of this subgenre is Dota 2, which is one of the most popular in the MOBA world. In recent months Dota has peaked at over 705 thousand concurrent players[2]. This popularity is due to the game dynamics that encourage players to engage for a long time [23].

In this context, artificial intelligence has been used to improve the gaming experience and game sales, for instance, through game recommendation in e-commerce stores [5], difficulty adjustment [19], intelligent agents [13], and in-game recommender systems [1, 4]. Regarding the latter, recommender systems address several problems in the context of video games, where some recommendations can help to improve the game, either as a strategy or as a development of new game features [2]. In particular, for the MOBA genre, the current methods [1, 25] increase the probability of winning a match and help beginners through the item recommendation based on the game context. However, to the best of our knowledge,

[1]https://www.statista.com/statistics/830090/mmo-moba-market-revenue/
[2]https://steamcharts.com/app/570

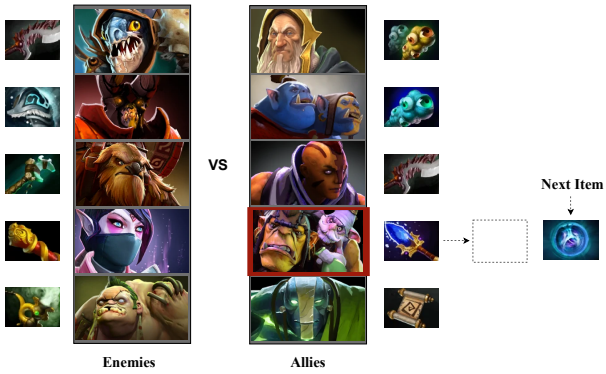


Figure 1: Example of recommending the next item to a hero based on the purchase history of the other participants.

no current method takes into account the sequential nature of the MOBA matches. In this kind of game, the purchase of an item is a decision that often depends on the game state at a specific time, that is, on the items previously purchased by the player and the items that each adversary buys at that time. In this sense, a sequential recommendation that considers the context of the moment of purchase would be relevant for gamers.

In this paper, we introduce a novel Hierarchical Transformer for Sequential Recommendation (HT4Rec). Inspired by previous work in the field of NLP that represent documents hierarchically [28], we model in-game item recommendation as a sequential problem with hierarchical representation. Our model consists of a group-aware contextual encoder, which models the relationship between a character and its enemies or allies to enrich its representations. In the upper level, HT4Rec employs a sequential encoder in order to capture the sequential behavior of the item consumption. Our method consistently outperforms the competitive baselines models GRU4Rec and SASRec, as well as the extended versions of these models that we propose and introduce in this article, GRU4Rec+ and SASRec+. The improvements are not only shown in the Dota 2 dataset but also in a movie recommendation dataset, demonstrating the further potential of the model. Moreover, the attention weights can be used as a form of interpretability through their visualization, increasing the reliability of the provided recommendations. In summary, our contributions are: (i) introducing HT4Rec which addresses in-game item recommendation as a sequential model, (ii) enriching the user representation by means of a hierarchical structure in the architecture, (iii) showing how HT4Rec attentions could be used to interpret model suggestions.

We share with the community HT4Rec source code as well as the Dota 2 dataset processed to train and test other eventual group-aware sequential recommendation approaches[3].

2 MOBA GAMES: OVERVIEW AND ITS SEQUENTIAL RECOMMENDATION PROBLEM

MOBA is a sub-genre of online video games with Dota 2 being one of the most famous. This game is about the confrontation of two teams *Radiant* and *Dire*. Each team consists of 5 players and, in turn, each player is represented by a character/hero [12]. The player chooses a single hero from a total of more than 110 available in the game. The choice often depends on the allied strategy to fight the enemy team [3]. During the match, players earn gold to purchase items from a group of over 180, helping the heroes unlock their maximum potential in-game. Proper itemization[4] is important, especially when faced with a situation that is not ideal for the specific hero one is using, increasing your chances of winning a duel or even the match. This presents opportunities for in-game recommendation systems that could enhance the gaming experience.

Several works focus on the suggestion of fixed sets of final/upgraded items, similar to recommendations that can be found on websites posted by experts. However, due to the inherent sequential nature of the game, a fixed item recommendation may not be sufficient as players have to purchase items multiple times in the game. Therefore, a good choice of items throughout a match could increase the chances of winning. A sequential recommendation approach would be helpful to address this need. However, current sequential methods are not specifically designed for the context of MOBA games. In this sense, a system should recommend items to a character based on the specific context of the match, specifically taking into account the other characters involved in the game and the items they possess at a specific time (see Figure 1). A system with these characteristics could be helpful not only for new players to reduce their learning curve but also for recurring players to obtain itemization alternatives improving the gaming experience for both.

3 RELATED WORK

3.1 In-game Recommendation for MOBA Games

In recent years, in-game recommendation methods for MOBA games have received increasing interest, where most works focused on character [4, 7, 16] and item [1, 11, 25] recommendation. In this work, we will focus on the latter. Item recommendation systems show two approaches based on machine learning methods. One for recommending an item for a specific character given an initial set of items [11] and another for the recommendation of a fixed itemset given a target character and his/her adversaries [1, 25]. Regarding the second approach, a fixed set of items is a valuable recommendation for a user previous to starting a match. However, a MOBA game is an inherently sequential process where it is desirable to obtain a recommendation at a specific time of the match. For that reason, unlike previous approaches that only use character features for itemset recommendations, we leverage meaningful sequential information about a match to recommend appropriate items sequentially.

3.2 Sequential Recommendation

The goal of sequential recommendation systems is to suggest items by modeling the sequential dependencies on user-item interactions in a sequence [26]. In recent years, deep learning techniques for sequence modeling are getting widely used in tackling sequential

[3]https://github.com/vgaraujov/HT4Rec-RecSysMOBA

[4]It is the action of acquiring items for a specific character.

recommendation issues. Early work proposed convolutional neural network (CNN) [21] and recurrent neural networks (RNN) [9] models for modeling item sequences. Then, more recently, the introduction of the Transformer model [24] and its success in several areas motivated its use in recommender systems. SASRec [10] uses a Transformer as an autoregressive model to predict future items. BERT4Rec [20] adopt the masked language modeling [6] formulation to predict random masked items in a sequence by jointly conditioning their left and right context. Based on these approaches, extensions have been proposed. For example, S3-Rec [29] for sequential recommendation self-supervised learning based on mutual information maximization, SGNN-HN [14] for session-based recommendation using a star graph neural network and SSE-PT [27] for personalized recommendations. However, none of them propose the use of hierarchical Transformers to model the user-item interactions in lower layers and the modeling of sequential dependencies in the upper layers. Previous work explored session-based recommendation with hierarchical recurrent neural networks [17] to model the users' activity across sessions observing their interests over time. This approach is not directly comparable with ours because the users are the MOBA characters, who can change their taste depending on the current game (e.g., depending on the assigned role). Therefore, we do not consider this as one of our baselines.

4 METHOD

Here, we introduce our approach for sequential item recommendation in MOBA games called HT4Rec, which stands for **H**ierachical **T**ransformers **f**or Sequential **R**ecommendation. Figure 2(a) shows the proposed architecture, which, unlike GRU4Rec (Figure 2(c)), is based on Transformer and which, unlike SASRec (Figure 2(b)), has a hierarchical structure and an explicit representation of the user. Before describing the model, we first introduce the research problem and the notations used in this paper.

4.1 Problem Statement

Let $\mathcal{U} = \{u_1, u_2, \ldots, u_{|\mathcal{U}|}\}$ be a set of users[5], $\mathcal{V} = \{v_1, v_2, \ldots, v_{|\mathcal{V}|}\}$ be a set of items, and $\mathcal{S}_u = [\mathbf{v}_1^{(\mathbf{u})}, \mathbf{v}_2^{(\mathbf{u})}, \ldots, \mathbf{v}_n^{(\mathbf{u})}]$ denotes a sequential list of items $v^{(u)} \in \mathcal{V}$ with which a user $u \in \mathcal{U}$ interacts. In the standard sequential recommendation, given an item $v_i^{(u)}$ of the interaction record $\mathcal{S}_u$, the objective is to predict the item with which the user u will interact in the sequence step $i+1$. This can be formalized as $p(v_{i+1}^{(u)}|v_{\leq i})$, the probability of recommending an item $v_{i+1}^{(u)}$ for user u given all previous items $v_{\leq i}$.

In the case of MOBA games, hero itemization not only depends on the tastes of the gamer but also on the needs of the team to combat enemies and allies, which can change over time. Therefore, enemies, allies and their itemization are the most important factors in deciding which items to acquire. We extend the standard sequential recommendation by explicitly using information of enemies, allies and their items to influence the target user representation which is used to model sequential recommendation. Under this scheme, user $u \in \mathcal{U}$ is the character to whom we want to recommend an item, $u^a \in \mathcal{U}^{(u^a)}$ is an ally hero and $u^e \in \mathcal{U}^{(u^e)}$ is an enemy in the game (there are usually 4 allies and 5 enemies).

[5]Users in this paper refer to character/hero selected by the player.

Based on this, we can compute the probability of recommending an item $v_{i+1}^{(u)}$ for a select u given the information of the allies u^a and enemies u^e and all their previously purchased items, formally as $p(v_{i+1}^{(u)}|u, v_{\leq i}^{(u)}, u^a, v_{\leq i}^{(u^a)}, u^e, v_{\leq i}^{(u^e)})$. We can simplify this expression to $p(v_{i+1}^{(u^a)}|c_{\leq i}^{(u)})$, where $\mathbf{c}^{(\mathbf{u})}$ is a group-aware contextual representation of u containing all the game information of previous steps.

4.2 Input Representation Layer

The goal of the input representation layer is to prepare a representation for the user u, allies $u^{(a)}$, and enemies $u^{(e)}$ and their consumed items. To do this, we took inspiration from previous work using learned embeddings to get explicit latent representations [21, 27]. Below we will explain how the input for the selected user u is computed. The same procedure applies to allies and enemies of the match.

First, we compute the user embedding $\mathbf{u}$ that represents a character that participates in the game. Note that in MOBA games, a character can be an ally or an enemy. Since the embedding vector $\mathbf{u}$ does not explicitly represent whether it is the allies or the enemy, we use a team embedding $\mathbf{t}$. These two embeddings are added together to generate the actual hero embedding. The team embedding may not be necessary for scenarios other than MOBA games (See Section 6 for an example), and for that reason, we do not explicitly include it in the Figure 2(a).

Then, we compute the item embeddings $\mathbf{v}^{(u)}$ that represent the items used by user u at a specific sequence step i. As the equation 1 shows, this embedding $\mathbf{v}^{(u)}$ is concatenated ($\oplus$) with the former to get the final input embedding $\mathbf{x}$, which has d dimensions. This is done for all match participants (1 target user, 4 allies and 5 enemies) to obtain the group representation $\mathbf{X}_i$ at step i.

$$\mathbf{x} = (\mathbf{u} + \mathbf{t}) \oplus \mathbf{v}, \ \textit{where } \mathbf{x} \in \mathbb{R}^d \tag{1}$$

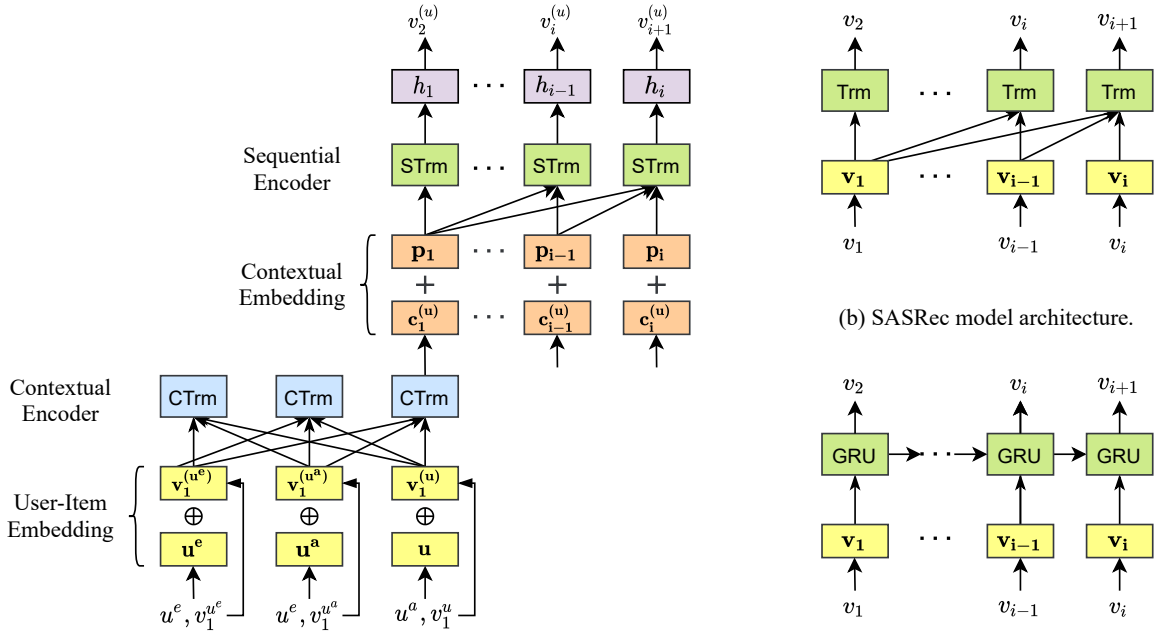
4.3 Contextual Transformer Layer

The aim of this layer is to obtain a group-aware contextual embedding of the target character at a specific time step i of the match. This embedding is influenced by the enemies, allies and their items. We use a bi-directional Transformer ($CTrm$), which has been used previously for this purpose [25], to compute interactions between allied and enemy characters of the match with the self-attention mechanism. The output is the contextual embedding of the user to whom items will be recommended $\mathbf{c}_i^{(u)}$ given the representation of all match participants $\mathbf{X}_i$.

$$\mathbf{c}_i^{(u)} = CTrm(\mathbf{X}_i), \ \textit{where } \mathbf{c}^{(u)} \in \mathbb{R}^d \tag{2}$$

4.4 Sequential Transformer Layer

This layer, inspired by SASRec [10], consists of a Transformer encoder for sequential modeling ($STrm$). The key difference with the contextual encoder is that this sequential encoder uses a mask that modifies the self-attention sub-layer to prevent positions from attending to subsequent positions. This mechanism ensures that the predictions for position **i** can depend only on the known context representations at positions less than **i** (see Figure 2(b)). Since the self-attention mechanism is unaware of the input positions, we use a learnable positional embedding $\mathbf{p}$ that is added to the context



(a) HT4Rec model architecture.

(b) SASRec model architecture.

(c) GRU4Rec model architecture.

Figure 2: Sequential recommendation model architectures. HT4Rec (a) learns an explicit contextual user representation to model sequential recommendation, while SASRec (b) and GRU4Rec (c) methods are only sequential recommendation models. $\oplus$ represents a concatenation operation. Trm, CTrm, and STrm refer to Transformer, Contextual Transformer and Sequential Transformer, respectively.

embedding $\mathbf{c}^{(u)}$ before feeding the sequential encoder. The output of this layer is a hidden state $\mathbf{h}$ for each position of the sequence.

$$\mathbf{h}_i = STrm(\mathbf{c}_i^{(u)} + \mathbf{p}_i), \ where \ \mathbf{h}_i \in \mathbb{R}^d \tag{3}$$

4.5 Recommendation Layer

After the hierarchical Transformers processing, $\mathbf{h}$ is obtained for all elements of the input sequence. Assuming that we have the representation $\mathbf{h}_i$ at time step i, we then predict the next item v_{i+1} for the ally character u^a. For this, we use a feed-forward network ($\mathbf{W}_{rec}$) along with a $softmax$ function to produce an output distribution over target items.

$$P(v_{i+1}|v_{\leq i}) = softmax(\mathbf{W}_{rec}\mathbf{h}_i) \tag{4}$$

5 EXPERIMENTS

5.1 Dataset and Preprocessing

We use a public dataset provided by Kaggle[6] consisting of 50k Dota 2 matches. An important feature of this dataset is the existence of the match purchase history. We did not test other datasets, such as the LoL dataset used in [25], because sequential consumption of in-game items is not available in the raw data, to the best of our knowledge. We had to perform preprocessing to use these data because the Dota 2 dataset does not provide a specific structure for a sequential recommender system. We first filter consumables from the full purchase history because they may not be situational

[6]https://www.kaggle.com/devinanzelmo/dota-2-matches

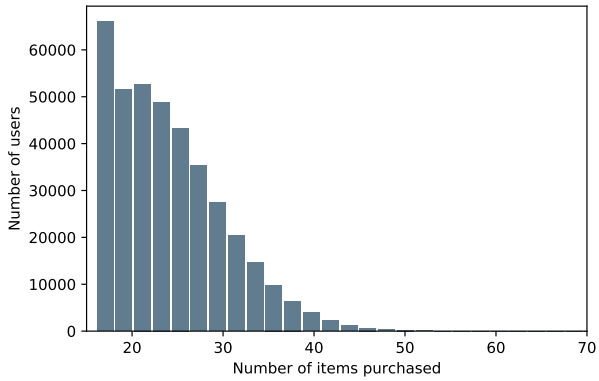


Figure 3: Distribution of the number of items purchased by each player.

and therefore unpredictable. For example, consumable items are typically purchased with the user's remaining gold, a variable not considered in this work. Also, we use the winning teams as ground truth based on the assumption that the winning users perform better itemization. We borrow these assumptions from previous work on in-game recommendation [1, 11, 25].

To build the dataset structure, we based it on the purchase history of each player. For this, first of all, we collect all purchases

throughout the match from a player whose team is the winner. Purchases made by a hero u are a sequence of items ordered by time, which we represent as $\mathcal{S}_u = [\mathbf{v}_1^{(\mathbf{u})}, \mathbf{v}_2^{(\mathbf{u})}, \ldots, \mathbf{v}_n^{(\mathbf{u})}]$. Then, we verify the purchases their enemies and allies made during each purchase by an ally, and we repeat this process for all historical purchases. Thus, each instance is the sequence of items purchased by the hero and the purchases of each of the enemies and allies. The sequence is based solely on the purchase time of the hero.

The dataset we use in our experiments consists of 46,065 matches, which result in 113 heroes, 180 items, and 230,325 instances. The data was randomly divided into two subsets for training (80%) and testing (20%), taking into account that in both all the characters must be present. Note that the dataset was divided by matches played and not by user instance, preventing information leakage of games from the training set to the test set. On the other hand, Figure 3 shows the distribution of the number of items purchased by users (sequence length). We find that the minimum and the maximum number of player purchases are 10 and 180, respectively. However, the distribution is densely concentrated between 10 and 60, being 22.6 the average of purchases with a $SD = 4.8$. For this reason, we set our model with a sequence length of $n = 60$ because it considers the largest number of matches played. Section 5.4 shows how the sequence length parameter influence the final performance of the model. As mentioned above, we make our dataset publicly available.

5.2 Baselines and Implementation Details

To verify the effectiveness of our method, HT4Rec, we compare it with two groups of recommendation baselines, non-sequential and sequential. The non-sequential methods are:

- **POP**: It is a simple baseline that ranks all items by popularity for each sequence of users. Popularity is determined by frequency.
- **ANN** [1]: It is a shallow artificial neural network proposed to recommend a set of elements given a MOBA character as input.
- **TTIR** [25]: It is a model based on Transformer proposed to compute the interactions between all the characters of a MOBA match. In this way, the resulting recommendations are contextual-aware, but not sequential.

The goal of using these methods is to evaluate how well a model that does not exploit sequential information performs in a sequential setup. Because this first set of baselines are not directly comparable to our approach, we adapted them for a sequential evaluation setup. To do this, we rank the list of scores produced for each item by POP, ANN, or TTIR. We use the sorted ranked items as a recommendation for each step of the sequence. In other words, the first ranked item is recommended at the first step, the second-ranked item is suggested at the second step, and so on.

The second group consists of methods that take advantage of the sequence information to produce recommendations, which are:

- **GRU4Rec** [9]: It is a session-based model, which uses a GRU network to capture sequential dependencies to make recommendations.
- **SASRec** [10]: A state-of-the-art model based on Transformer conditioning to capture dependencies from left to right of user sessions for sequential recommendation.

This second group of sequential baselines is comparable to our approach. However, these models induce an implicit user representation from embeddings of visited items, while HT4Rec learns an explicit user embedding representing user preferences at each step of a sequence. For a more direct and fair comparison, we also include an extended version of these models that use explicit representations of the user concatenated with the classic input representation of items. We will refer to these versions as **GRU4Rec+** and **SASRec+**. We do not consider BERT4Rec as a baseline because our method is based on predicting future elements and not on masked modeling.

We implement HT4Rec using PyTorch [15]. As default hyperparameters, we set the contextual encoder layer number $l_1 = 1$, the sequential encoder layer number $l_2 = 1$, the attention head number $h = 4$, the dimensionality $d = 512$ and the maximum sequence length $n = 60$. We train our model for 50 epochs using the Adam optimizer with learning rate of 2e-4, $\beta_1 = 0.9$, $\beta_1 = 0.98$ and weight decay of 1e-4. We adopt the *CrossEntropy* loss as the objective function. For a fair and consistent comparison, we use the same setup to train all the sequential baselines.

5.3 Results

To evaluate our model, we consider Precision@k, Recall@k, and Mean Reciprocal Rank (MRR@k) metrics with k=3,5,10.

As shown in Table 1, those baselines that are non-sequential models do not achieve competitive performance. This confirms the importance of considering the sequential nature of MOBA games when designing in-game recommendation systems. On the other hand, HT4Rec achieved consistently better results compared to the state-of-the-art sequential methods. Our model shows an increase percentage of 4.24% at Precision@10, 4.20% at Recall@10, and 10.04% at MRR@10 with respect to the best sequential baseline SASRec. These gaps are even larger in the @5 and @3 metrics. Regarding our runner up baseline SASRec+, HT4Rec also shows an increase percentage of 1.38% at Precision@10, 1.37% at Recall@10, and 5.15% at MRR@10. These results prove that our model can recommend relevant items in the top ranking positions with more confidence. This is very significant due to the difficulty of the task.

Also, the statistical test shows HT4Rec is consistently better than the other methods. The effect size on the different metrics varies from medium (0.68 - 0.74) when comparing HT4Rec against GRU4Rec and SASRec, to small effect size (0.18 - 0.33) when comparing against GRU4Rec+ and SASRec+.

Furthermore, GRU4Rec [9] and SASRec [10] perform worse than their mightier version (SASRec+ and GRU4Rec+). This demonstrates the importance of using an explicit representation of the user to model the game's characters. However, HT4Rec outperforms SASRec+ and GRU4Rec+, demonstrating the importance of hierarchical modeling that focuses on capturing relationships between users and then sequential relationships.

In summary, the results show the powerful ability of this Transformer-based model to include relevant contextual and hierarchical information and then recommend items sequentially.

Table 1: Performance comparison of different methods on next-item prediction. The best performing method is in bold, and the runner up method is underlined. The + symbol in model names indicates the use of explicit representation of the user in the input. Improvement over two baselines is shown in the last columns. HT4Rec is statistically significantly better than all the other methods with $p < 2e^{-16}$, using the Bonferroni correction for multiple comparisons.

	POP	ANN	TTIR	GRU4Rec	SASRec	GRU4Rec+	SASRec+	HT4Rec	Improv. vs.	
	(a)	(b)	(c)	(d)	(e)	(f)	(g)	(h)	(h)-(e)	(h)-(g)
Precision@3	0.0038	0.0313	0.1279	0.2553	0.2558	0.2666	<u>0.2688</u>	**0.2798**	9.38%	4.09%
Recall@3	0.0114	0.0871	0.3838	0.7661	0.7675	0.7999	<u>0.8063</u>	**0.8394**	9.37%	4.11%
MRR@3	0.0094	0.0599	0.2536	0.6532	0.6534	0.6843	<u>0.6879</u>	**0.7275**	11.34%	5.76%
Precision@5	0.0025	0.0271	0.0997	0.1670	0.1680	0.1736	<u>0.1750</u>	**0.1801**	7.20%	2.91%
Recall@5	0.0125	0.1357	0.4988	0.8353	0.8390	0.8683	<u>0.8751</u>	**0.9006**	7.34%	2.91%
MRR@5	0.0096	0.0646	0.2802	0.6691	0.6702	0.6999	<u>0.7033</u>	**0.7416**	10.65%	5.45%
Precision@10	0.0013	0.0222	0.0517	0.0915	0.0919	0.0939	<u>0.0945</u>	**0.0958**	4.24%	1.38%
Recall@10	0.0127	0.2228	0.5174	0.9152	0.9190	0.9390	<u>0.9447</u>	**0.9576**	4.20%	1.37%
MRR@10	0.0097	0.0760	0.2833	0.6799	0.6810	0.7096	<u>0.7127</u>	**0.7494**	10.04%	5.15%

5.4 Ablation Study

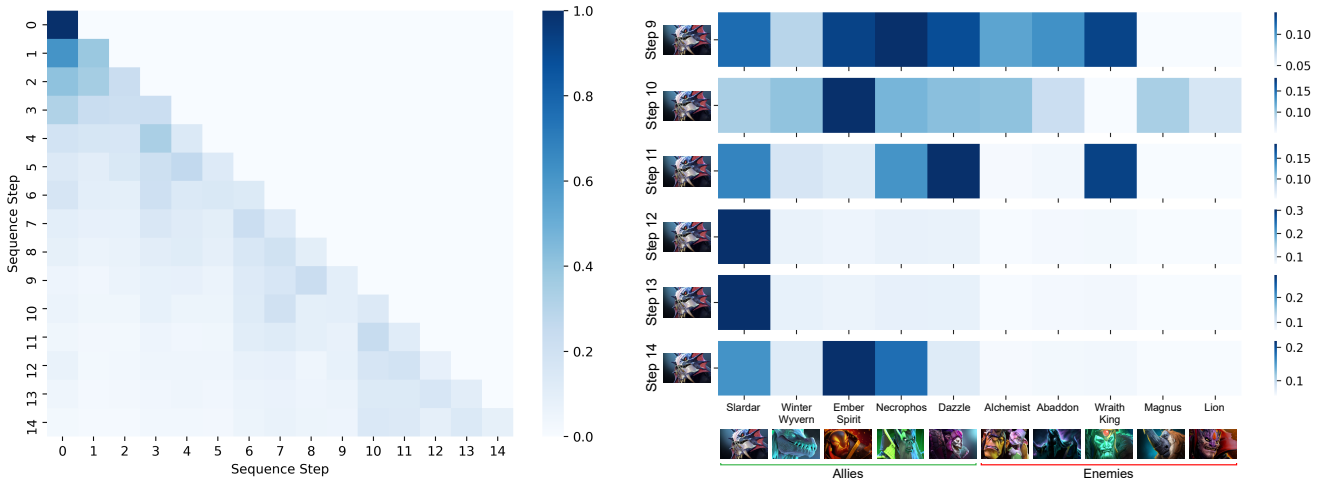
To better understand the impact of the main components of HT4Rec, we performed ablation experiments. We focus on some hyperparameters, such as the number of layers (l_1, l_2), the number of attention heads (h), and the loss function used to optimize the model. Additionally, we evaluate by removing positional embedding and user embedding, which are two key components of our model. Table 2 shows the performance of our default method ($l_1 = 1$, $l_2 = 1$, $h = 4$, $loss = CrossEntropy$) and the proposed variants. Below we specify and analyze the variations made to the model:

- *Number of layers*: HT4Rec has two types of Transformer encoders that can have an independent number of layers. By setting $l_1 = 2, l_2 = 1$ and $l_1 = 1, l_2 = 2$, the model exceeds the default model ($l_1 = 1, l_2 = 1$). When $l_1 = 1, l_2 = 2$, performance increases considerably more. These results suggest that the contextual encoder learns to model more user-element interaction patterns using more layers. On the other hand, increasing the number of sequential Transformer layers seems to be useful for learning more complex transition patterns from a match.
- *Number of attention heads*: HT4Rec uses h as a shared hyperparameter for contextual and sequential encoders. In other words, both encoders use the same number of heads but not the same learnable weights. By reducing $h = 2$, the model performance decreases. In contrast, by increasing $h = 8$, the model increases performance. These results are consistent with previous work stating that a larger size of h is useful for capturing long-distance dependencies [10].
- *Loss function*: We also explored the behavior of HT4Rec using *BPR* [18] and *TOP*1 [9] loss functions. None of the loss functions increase the performance of the model, suggesting that our approach does not benefit from pairwise-ranking losses. This may be related to GRU4Rec results [9] where it is shown that these losses perform well with a higher dimensional model.
- *Remove positional embedding* **p**: Positional encoding is an essential part of Transformer that injects information about the position of elements in the sequence [24]. As in previous works [20], we found that model performance is slightly reduced in our Dota 2 dataset due to the short sequences.
- *Remove user embedding* **u**: By removing this embedding, model performance drops dramatically, confirming that our model takes advantage of explicit user representation to provide an appropriate recommendation.
- *Remove allies or enemies*: This last experiment consists of eliminating information from allies or enemies of the match. The results show that the model reduces its performance due to the lack of such contextual information.

Additionally, we explore the impact on the performance of the model by varying the maximum sequence length n used and hidden dimensionality d for training. Table 3 shows the performance results of these experiments. On the one hand, we found that increasing the maximum sequence length boosts the performance of the model. However, the increase is slight between $n = 40$ and $n = 100$. This is because the Dota 2 dataset is a short sequence dataset with an average of ~ 22 and a maximum of 180 in sequence length. For this reason, the model performance achieves its best result with $n = 40$ and $n = 60$, and it manages to improve the results of the longest minority instances of the dataset when $n = 80$ and $n = 100$. On the other hand, regarding the hidden dimensionality of the model, the result is consistent with previous work [20] that states the performance of the model tends to converge as the dimensionality increases. Model performance is negatively affected when the dimensionality is too low, suggesting that the model cannot capture relevant patterns to solve the recommendation task.

5.5 What is Multi-head Attention Capturing?

The goal of this section is to reveal whether HT4Rec is learning meaningful patterns from the data by visualizing the average of the attention weights. Our approach has an encoder to produce contextual user-item representations and a sequential encoder to model the sequential interactions of the match. Therefore, we analyze the attentions of each encoder to discover if the hierarchical structure affects the patterns that the model captures. To do this, we randomly select an example from the test suite, the results are shown and discussed below.



(a) Sequential encoder attention weights.

(b) Contextual encoder attention weights.

Figure 4: Visualization of the attention weights of (a) sequential encoder and (b) contextual encoder. A darker color means more attention, while a lighter color means no attention.

Table 2: Variation of main components of HT4Rec. Default model is $l_1 = 1$, $l_2 = 1$, $h = 4$, $loss = CrossEntropy$.

	Precision@5	Recall@5	MRR@5
Default	0.1801	0.9006	0.7416
$l_1 = 2, l_2 = 1$	0.1805	0.9024	0.7430
$l_1 = 1, l_2 = 2$	0.1803	0.9014	0.7416
$l_1 = 2, l_2 = 2$	0.1809	0.9046	0.7460
$h = 2$	0.1797	0.8986	0.7373
$h = 8$	0.1803	0.9017	0.7419
$loss = BPR$	0.1756	0.8779	0.6984
$loss = TOP1$	0.1721	0.8607	0.6656
w/o **p**	0.1798	0.8990	0.7364
w/o **u**	0.1733	0.8667	0.7059
w/o allies	0.1785	0.8924	0.7311
w/o enemies	0.1796	0.8981	0.7378

Table 3: Impact of maximum sequence length n and hidden dimensionality d in HT4Rec.

n	20	40	60	80	100
Precision@5	0.1797	0.1800	0.1801	0.1801	0.1800
Recall@5	0.8985	0.9004	0.9006	0.9004	0.9002
MRR@5	0.7283	0.7397	0.7416	0.7401	0.7396
d	128	256	512	768	1024
Precision@5	0.1789	0.1797	0.1801	0.1802	0.1802
Recall@5	0.8947	0.8984	0.9006	0.9011	0.9012
MRR@5	0.7344	0.7386	0.7416	0.7406	0.7411

Figure 4(a) shows the resulting attention weights of the sequential encoder. The attentions are influenced by the subsequent mask that restricts the model to depend only on previous information and not on future information; thus, the upper diagonal of the heat map is always null. Furthermore, we note that HT4Rec attends not only on the closest past steps but also to the most distant past steps. This result differs from the results shown in previous work (SASRec), where the attention is uniformly distributed over the previous steps [10]. Due to the nature of this game, we believe that the model attends in a non-uniform way leveraging important aspects located in different parts (previous steps) of the match to recommend items suitable for the current specific context.

Figure 4(b) shows the visualization of the attentions from the contextual encoder. There are weights of attention for each element of the whole sequence, so we only show the visualization from the final steps (step 9 to step 14). In this example, *Slardar* is the user to whom we want to recommend items, while the rest of the characters are the allies and enemies that influence the contextual representation of that user. We found that attention is most intense when the user needs an item to counter a specific adversary or support a partner. Also, very often, when an enemy or ally acquires new items. In this match, *Slardar* is an off-lane utility hero that plays alongside *Winter Wyvern* against *Wraith King* and *Abaddon*. In steps 9 and 10, we observed that *Slardar* paid attention to most of the game participants. This is because the recommended element in step 10 is a *Force Staff*, which is good to engage or disengage a team fight. In steps 11, we observed that *Slardar* is heavily influenced by *Wraith King*, so the recommendation focused on countering that character. In steps 12 and 13, there is only strong self-attention. This is because none of the enemies bought an item during that period. So *Slardar* just focuses on its elements to figure out what the next final item will be. Finally, step 14 shows a distributed attention over the allies. This is because the recommendation at this time was *Lotus Orb*, which is a very useful shield to avoid the crowd control abilities the enemies have. Especially *Ember Spirit* benefits a lot from this item because it is very prone to stunning or slowing spells, and that would explain the increased attention paid to it.

In general, we find several patterns that show that the representation of a user is influenced by the presence of enemies or allies and by their acquired items. These results are consistent with previous work in the MOBA domain, showing that enemy information positively influences item recommendations for a character [25]. On the other hand, we argue that this visualization and analysis would serve as a form of interpretability. This opens an additional challenge to this kind of system for further research.

6 FURTHER IMPACT: HT4REC ON OTHER DOMAINS

The HT4Rec model captures user preferences through interaction with other users and their items at a specific time and the sequential relationships that occur across a whole sequential event. In this paper, we focus on the scenario of item recommendation in MOBA games, which is an ideal task for this architecture. However, the specialization of our model makes it suitable for other sequential group-aware item recommendation scenarios. An example of use is in user groups in a social network. In that case, the interaction between a user and the other group members could predict the next action or item to recommend based on their preferences. Another possible use case could be in video streaming services like Netflix, where multiple users share the same account. Contextual recommendations could be generated to a user based on the movies or series previously consumed by him/her and the other users of the account. Finally, we remark that HT4Rec architecture allows visualizing the attention weights to understand how the input characteristics are distributed both among users and in sequential steps. We argue that these features may serve as a step towards more transparent and interpretable recommendations.

6.1 HT4Rec for Sequential Movie Recommendation

To evaluate one of the possible scenarios mentioned above, we conducted an experiment with the MovieLens 1m dataset [8] to simulate Netflix accounts with multiple users. We created a dataset that contains profiles of shared video streaming accounts, in which users were grouped by extracting their geographical location based on their reported zip codes. Since video streaming accounts are generally shared by people living in the same or close households, users from the original MovieLens dataset were assigned a simulated account number shared by users that are geographically closest to them. Under this process, 1,069 groups where generated, containing 5 users each. The movies rated with a score of 3 or more by these users where then used as the contextual information used by the HT4Rec model.

To use HT4Rec for a movie recommendation task, we use the same method but remove the concept of teams. So our final architecture does not include the team embedding. Table 4 reports the performance of the models under this experiment. As can be seen, HT4Rec reports the best performance surpassing SASRec and slightly outperforming SASRec+. These results indicate that the contextual information given by the geographically-based groups is being used to adjust the recommendations. It can then be assumed that by exploring different approaches to the process of grouping users, different contextual information may be gathered by them, which could lead to improved recommendations.

Table 4: Performance comparison of different methods on movie recommendations on the geographical-based dataset. The best performing method is in bold, and the runner-up method is underlined.

	GRU4Rec	SASRec	GRU4Rec+	SASRec+	HT4Rec
Precision@3	0.1753	0.3179	0.1917	<u>0.3225</u>	**0.3241**
Recall@3	0.5259	0.9538	0.5750	<u>0.9674</u>	**0.9724**
MRR@3	0.4000	0.8903	0.4543	<u>0.9255</u>	**0.9304**
Precision@5	0.1275	0.1945	0.1341	<u>0.1961</u>	**0.1971**
Recall@5	0.6374	0.9740	0.6707	<u>0.9804</u>	**0.9856**
MRR@5	0.4254	0.8950	0.4762	<u>0.9285</u>	**0.9334**
Precision@10	0.0774	0.0988	0.0779	<u>0.0991</u>	**0.0994**
Recall@10	0.4438	0.9888	0.4909	<u>0.9912</u>	**0.9941**
MRR@10	0.9347	0.8971	0.7096	<u>0.9300</u>	**0.9347**

7 CONCLUSIONS

In this work, we proposed HT4Rec, a Transformer-based model with a hierarchical structure for next item recommendations. HT4Rec comprises two consecutive stages of processing, the first in charge of generating group-aware user-element representations and the second in charge of modeling the sequential dependencies to predict the next item. Experimental results on a Dota 2 dataset show that our model exceeds both non-sequential and sequential state-of-the-art baselines. One limitation of our work is that we were not able to evaluate other popular MOBA games such as League of Legends because public datasets, to the best of our knowledge, do not report the sequence in which items were consumed. Nevertheless, we demonstrated the usability of our model in other scenarios than video games obtaining consistent results. Additional experiments also show that this model actually takes advantage of the hierarchical structure to make suitable item recommendations in MOBA games. Among the ideas for future work, we considered exploring the attention weights as a way to explain the recommendations provided to the players. We also plan to conduct a user study to compare the explanations we have reported for HT4Rec against explanations of other attention-based models such as TTIR [25]. In this way, we could understand whether these attention weights and the way of visualizing them are actually more interpretable for users, an important aspect to drive trust in the recommendation system [22]. Such user study could also allow us to get qualitative information about the user experience with this type of character item recommendation in a regular gaming session. Finally, another work direction is to test our sequential hierarchical transformer model on additional domains where the sequential context has great potential.

ACKNOWLEDGMENTS

The main author thanks Jorge Esteban Martínez for insightful discussions about the DOTA game. This work was funded by ANID, Millennium Science Initiative Program, Code ICN17_002 (IMFD), as well as by ANID Basal funds, the National Center of Artificial Intelligence (CENIA), code FB210017.

REFERENCES

[1] Vladimir Araujo, Felipe Rios, and Denis Parra. 2019. Data Mining for Item Recommendation in MOBA Games. In *Proc. of the 13th ACM Conference on Recommender Systems* (Copenhagen, Denmark) *(RecSys '19)*. ACM, New York, NY, USA, 393–397. https://doi.org/10.1145/3298689.3346986

[2] Paul Bertens, Anna Guitart, Pei Pei Chen, and Africa Perianez. 2018. A machine-learning item recommendation system for video games. In *2018 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 1–4.

[3] O. Cavadenti, V. Codocedo, J. Boulicaut, and M. Kaytoue. 2016. What Did I Do Wrong in My MOBA Game? Mining Patterns Discriminating Deviant Behaviours. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*. 662–671. https://doi.org/10.1109/DSAA.2016.75

[4] Zhengxing Chen, Truong-Huy D Nguyen, Yuyu Xu, Christopher Amato, Seth Cooper, Yizhou Sun, and Magy Seif El-Nasr. 2018. The art of drafting: A Team-Oriented Hero Recommendation System for Multiplayer Online Battle Arena Games. In *Proc. of the 12th ACM Conference on Recommender Systems - RecSys '18*. ACM Press. https://doi.org/10.1145/3240323.3240345

[5] Germán Cheuque, José Guzmán, and Denis Parra. 2019. Recommender systems for Online video game platforms: The case of STEAM. In *Companion Proc. of The 2019 World Wide Web Conference*. 763–771.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *Proc. of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, 4171–4186. https://doi.org/10.18653/v1/N19-1423

[7] D. Gourdeau and L. Archambault. 2020. Discriminative neural network for hero selection in professional Heroes of the Storm and DOTA 2. *IEEE Transactions on Games* (2020), 1–1.

[8] F Maxwell Harper and Joseph A Konstan. 2015. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)* 5, 4 (2015), 1–19.

[9] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2016. Session-based Recommendations with Recurrent Neural Networks. In *ICLR 2016*.

[10] W. Kang and J. McAuley. 2018. Self-Attentive Sequential Recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. 197–206. https://doi.org/10.1109/ICDM.2018.00035

[11] W. Looi, M. Dhaliwal, R. Alhajj, and J. Rokne. 2018. Recommender System for Items in Dota 2. *IEEE Transactions on Games* (2018), 1–1. https://doi.org/10.1109/TG.2018.2844121

[12] Marçal Mora-Cantallops and Miguel-Ángel Sicilia. 2018. MOBA games: A literature review. *Entertainment computing* 26 (2018), 128–138.

[13] OpenAI, :, Christopher Berner, Greg Brockman, Brooke Chan, Vicki Cheung, Przemysław Dębiak, Christy Dennison, David Farhi, Quirin Fischer, Shariq Hashme, Chris Hesse, Rafal Józefowicz, Scott Gray, Catherine Olsson, Jakub Pachocki, Michael Petrov, Henrique Pondé de Oliveira Pinto, Jonathan Raiman, Tim Salimans, Jeremy Schlatter, Jonas Schneider, Szymon Sidor, Ilya Sutskever, Jie Tang, Filip Wolski, and Susan Zhang. 2019. Dota 2 with Large Scale Deep Reinforcement Learning. arXiv:1912.06680 [cs.LG]

[14] Zhiqiang Pan, Fei Cai, Wanyu Chen, Honghui Chen, and Maarten de Rijke. 2020. Star Graph Neural Networks for Session-Based Recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. Association for Computing Machinery, New York, NY, USA, 1195–1204. https://doi.org/10.1145/3340531.3412014

[15] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (Eds.), Vol. 32. Curran Associates, Inc. https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf

[16] I. Porokhnenko, P. Polezhaev, and A. Shukhman. 2019. Machine Learning Approaches to Choose Heroes in Dota 2. In *2019 24th Conference of Open Innovations Association (FRUCT)*. 345–350.

[17] Massimo Quadrana, Alexandros Karatzoglou, Balázs Hidasi, and Paolo Cremonesi. 2017. Personalizing Session-Based Recommendations with Hierarchical Recurrent Neural Networks. In *Proceedings of the Eleventh ACM Conference on Recommender Systems* (Como, Italy) *(RecSys '17)*. Association for Computing Machinery, New York, NY, USA, 130–137. https://doi.org/10.1145/3109859.3109896

[18] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence* (Montreal, Quebec, Canada) *(UAI '09)*. AUAI Press, Arlington, Virginia, USA, 452–461.

[19] Mirna Paula Silva, Victor do Nascimento Silva, and Luiz Chaimowicz. 2017. Dynamic difficulty adjustment on MOBA games. *Entertainment Computing* 18 (Jan. 2017), 103–123. https://doi.org/10.1016/j.entcom.2016.10.002

[20] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management* (Beijing, China) *(CIKM '19)*. Association for Computing Machinery, New York, NY, USA, 1441–1450. https://doi.org/10.1145/3357384.3357895

[21] Jiaxi Tang and Ke Wang. 2018. Personalized Top-N Sequential Recommendation via Convolutional Sequence Embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining* (Marina Del Rey, CA, USA) *(WSDM '18)*. Association for Computing Machinery, New York, NY, USA, 565–573. https://doi.org/10.1145/3159652.3159656

[22] Nava Tintarev and Judith Masthoff. 2011. Designing and evaluating explanations for recommender systems. In *Recommender systems handbook*. Springer, 479–510.

[23] April Tyack, Peta Wyeth, and Daniel Johnson. 2016. The Appeal of MOBA Games: What Makes People Start, Stay, and Stop. In *Proc. of the 2016 Annual Symposium on Computer-Human Interaction in Play* (Austin, Texas, USA) *(CHI PLAY '16)*. ACM, New York, NY, USA, 313–325. https://doi.org/10.1145/2967934.2968098

[24] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *31st Conference on Neural Information Processing Systems* (Long Beach, CA, USA) *(NIPS '17)*.

[25] Andrés Villa, Vladimir Araujo, Francisca Cattan, and Denis Parra. 2020. Interpretable Contextual Team-Aware Item Recommendation: Application in Multiplayer Online Battle Arena Games. In *Fourteenth ACM Conference on Recommender Systems* (Virtual Event, Brazil) *(RecSys '20)*. Association for Computing Machinery, New York, NY, USA, 503–508. https://doi.org/10.1145/3383313.3412211

[26] Shoujin Wang, Liang Hu, Yan Wang, Longbing Cao, Quan Z Sheng, and Mehmet Orgun. 2019. Sequential recommender systems: challenges, progress and prospects. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*. AAAI Press, 6332–6338.

[27] Liwei Wu, Shuqing Li, Cho-Jui Hsieh, and James Sharpnack. 2020. SSE-PT: Sequential Recommendation Via Personalized Transformer. In *Fourteenth ACM Conference on Recommender Systems* (Virtual Event, Brazil) *(RecSys '20)*. Association for Computing Machinery, New York, NY, USA, 328–337. https://doi.org/10.1145/3383313.3412258

[28] Xingxing Zhang, Furu Wei, and Ming Zhou. 2019. HIBERT: Document Level Pre-training of Hierarchical Bidirectional Transformers for Document Summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, 5059–5069. https://doi.org/10.18653/v1/P19-1499

[29] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. 2020. S3-Rec: Self-Supervised Learning for Sequential Recommendation with Mutual Information Maximization. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management* (Virtual Event, Ireland) *(CIKM '20)*. Association for Computing Machinery, New York, NY, USA, 1893–1902. https://doi.org/10.1145/3340531.3411954