Attribute-aware Diversification for Sequential Recommendations

Anton Steenvoorden[1,2] Emanuele Di Gloria[2] Wanyu Chen[1,3]
Pengjie Ren[1] Maarten de Rijke[1,2]
[1]University of Amsterdam, Amsterdam, The Netherlands. [2]Ahold Delhaize, Zaandam, The Netherlands.
[3]National University of Defense Technology, Changsha, China
{anton.steenvoorden,emanuele.di.gloria}@aholddelhaize.com,{w.chen2,p.ren,m.derijke}@uva.nl

ABSTRACT

Users prefer diverse recommendations over homogeneous ones. However, most previous work on Sequential Recommenders does not consider diversity, and strives for maximum accuracy, resulting in homogeneous recommendations. In this paper, we consider both accuracy and diversity by presenting an Attribute-aware Diversifying Sequential Recommender (ADSR). Specifically, ADSR utilizes available attribute information when modeling a user's sequential behavior to simultaneously learn the user's most likely item to interact with, and their preference of attributes. Then, ADSR diversifies the recommended items based on the predicted preference for certain attributes. Experiments on two benchmark datasets demonstrate that ADSR can effectively provide diverse recommendations while maintaining accuracy.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

KEYWORDS

Sequential recommendation, Attribute-aware diversification

1 INTRODUCTION

Recommender Systems (RSs) are widely used to help users find items they are interested in. Traditional RSs such as Collaborative Filtering and Matrix Factorization [9] provide recommendations using a decomposition of a user-item interaction matrix. These approaches do not take the order of interactions into account and fail to capture the users' evolving preferences. Sequential Recommenders (SRs) have attracted a lot of attention, as they can exploit the order of interactions [8]. In industry, SRs allow e-commerce retailers to provide recommendations to users, based on their sequence of interactions and have proven to be very effective [6–8].

Side information in SRs, e.g., item attributes (category, genre, etc.), has proven to be useful for capturing user preferences. E.g., Bai et al. [2] use available item attributes to make their model attribute-aware by obtaining a unified representation from items and their attributes. And Chen et al. [5] use item attributes to infer users' future intention, allowing them to do intent-aware recommendation. However, the focus of these methods lies on increasing performance in terms of accuracy only, resulting in homogeneous recommendations. Moreover, diversity is an important metric to consider as it has been shown that users prefer diverse search results opposed to highly accurate, but redundant ones [3, 10].

We propose to use item-attribute information to diversify SRs. For instance, if we know that a user enjoys *documentaries* and *thrillers*, and this user is in search of films about "Artificial Intelligence" (AI), they are likely interested in both a *documentary* about AI and in a *thriller* where general AI achieves world domination. Genre information and preferences can be used to present diverse recommendations related to AI, collectively covering multiple genres. To this end, we present an Attribute-aware Diversifying Sequential Recommender (ADSR), which considers both accuracy and diversity while generating the list of recommendations.

The ADSR consists of three modules: an Attribute-aware Encoder (AE), an Attribute Predictor (AP), and an Attribute-aware Diversifying Decoder (ADD). The AE models the sequence of item interactions and the sequence of item-attributes. The AP learns and predicts the user's preference on attribute level. Finally, the ADD incrementally generates diversified recommendations by using the predictions from the AP and trading off accuracy and diversity.

The AE and AP both make predictions and are therefore optimized in a multi-task learning paradigm [4]. We carry out experiments on two benchmark datasets. The results demonstrate that ADSR benefits from modeling and predicting item attributes, and can provide attribute-aware diversified recommendations while preserving accuracy.

To sum up, the contributions of this work are as follows:

- We propose ADSR, which is one of the first to address diverse recommendations for SRs.
- We devise AP and ADD modules to generate attribute-aware diversified recommendations by jointly optimizing item recommendation and item attribute prediction as an auxiliary task.

2 Attribute-aware Diversifying Sequential Recommender (ADSR)

Given user u with behavior sequence $S_v = \{v_1, \ldots, v_t, \ldots, v_T\}$ and corresponding item attribute sequence $S_c = \{c_1, \ldots, c_t, \ldots, c_T\}$, where v_t is the item u interacts with at time step t, and c_t is the attribute of v_t, we aim to create a diversified list of recommendations R_L. Formally, let $P(c_j \mid S_c, S_v)$ be the importance of c_j based on the sequences and let $P(R_L \mid S_v, S_c, c_j)$ be the probability of recommending R_L conditioned that c_j is the user's preferred attribute.

AIIS'20, July 30, 2020, Virtual Event, China

ACM ISBN TBA.
https://doi.org/TBA

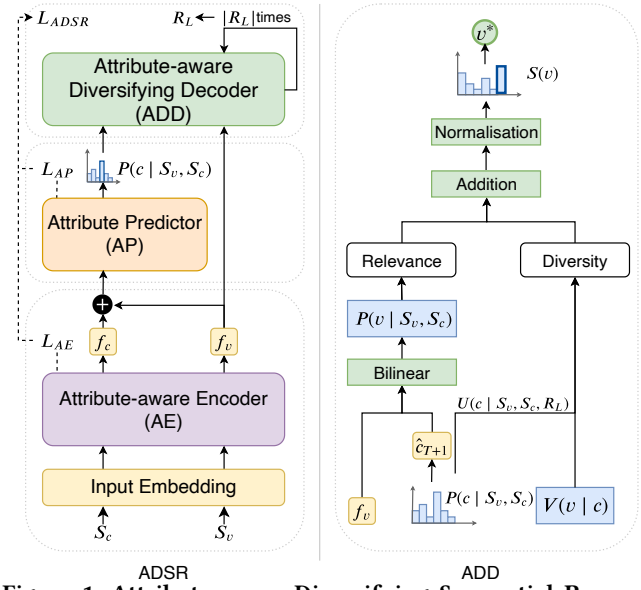

Figure 1: Attribute-aware Diversifying Sequential Recommender (ADSR) overview. The input to the recommender are item and attribute sequences. The output is a diversified list of recommendations based on learned attribute preferences.

Then, we find R_L by maximizing $P(R_L \mid S_v, S_c)$, defined as:

$$P(R_L \mid S_v, S_c) = \sum_{j=1}^{|C|} P(R_L \mid S_v, S_c, c_j) P(c_j \mid S_c, S_v), \tag{1}$$

Optimising $P(R_L \mid S_v, S_c)$ directly is difficult due to the large search space [1]. Therefore, we propose to generate R_L iteratively, by appending the item with the highest score $S(v_i)$ to R_L at each time step, similar to [1, 3]:

$$S(v_i) = \lambda_s \cdot S_{\text{rel}}(v_i) + (1 - \lambda_s) \cdot S_{\text{div}}(v_i), \tag{2}$$

where $S_{\text{rel}}(v_i)$ is the relevance score for item v_i (see §2.1), and $S_{\text{div}}(v_i)$ is the attribute-aware diversity score (see §2.3). The hyper-parameter λ_s is used to balance S_{rel} and S_{div}, giving control over the accuracy and diversity of the trained model.

We propose ADSR to model $S_{\text{rel}}(v_i)$ and $S_{\text{div}}(v_i)$, as shown in Figure 1. The Attribute-aware Encoder (AE) models the input sequences S_c, S_v to get $\mathbf{h}_c$ and $\mathbf{h}_v$, respectively. Next, the Attribute Predictor (AP) predicts the next item attribute distribution $P(\mathbf{c} \mid S_v, S_c)$. Finally, the Attribute-aware Diversifying Decoder (ADD) uses these outputs to generate the diversified list R_L.

2.1 Attribute-aware Encoder

First, the sequence of attributes is encoded using a bidirectional RNN with GRU-cells [7]. The input of this GRU is a sequence of item attribute embeddings concatenated with the item embeddings: $\{\mathbf{p}_1, \mathbf{p}_2, \ldots, \mathbf{p}_T\}$ where $\mathbf{p}_t = [\mathbf{c}_t; \mathbf{v}_t] \in \mathbb{R}^{d_c + d_v}$. The output of the GRU are hidden representations $\{\mathbf{h}_{c_1}, \mathbf{h}_{c_2}, \ldots, \mathbf{h}_{c_T}\}$ where $\mathbf{h}_{c_t} \in \mathbb{R}^{2d_{\text{GRU}}}$. Then, a second bidirectional RNN is used to get the representation for the sequence of item interactions. The input to this encoder is the concatenation of the item embeddings and the hidden states: $\mathbf{q}_t = [\mathbf{v}_t; \mathbf{h}_{c_t}]$, yielding $\{\mathbf{h}_{v_1}, \mathbf{h}_{v_2}, \ldots, \mathbf{h}_{v_T}\} \in \mathbb{R}^{2d_{\text{GRU}}}$.

Next, we employ additive attention to get the global preference $\mathbf{f}_v$ and $\mathbf{f}_c$, capturing the complete sequences, as follows:

$$\begin{aligned} \mathbf{f}_v &= \sum_{j=1}^{T} \alpha_{Tj} \mathbf{h}_{v_t} \\ \alpha_{Tj} &= \text{softmax}\left(\mathbf{W}_p \tanh(\mathbf{W}_q \mathbf{h}_{v_T} + \mathbf{W}_k \mathbf{h}_{v_j})\right), \end{aligned} \tag{3}$$

where matrices $\mathbf{W}_q, \mathbf{W}_k$ are used to transform $\mathbf{h}_i$ into a latent space, and $\mathbf{W}_p \in \mathbb{R}^{2d_{\text{GRU}} \times 1}$ is used to get the attention weights. $\mathbf{f}_c$ is calculated as in Eq. 3 using the attention weights α_{Tj} obtained for $\mathbf{f}_v$. Finally, AE calculates the relevance score as:

$$S_{\text{rel}}(v_i) = P(v_i \mid S_v, S_c) = \frac{\exp(\mathbf{g}^T \mathbf{v}_i)}{\sum_{j=1}^{|V|} \exp(\mathbf{g}^T \mathbf{v_j})}, \tag{4}$$

where $\mathbf{g} = [\mathbf{f}_v; \hat{\mathbf{c}}_{T+1}]\mathbf{W}_g \in \mathbb{R}^{d_v}$ and $\hat{\mathbf{c}}_{T+1} = \sum_{j=1}^{|C|} P(c_j \mid S_v, S_c) \cdot \mathbf{c}_j$ is the weighted average item attribute based on AP's predicted attribute preference. For clarity, the loss induced by AE (L_{AE}) is described in Section 2.4.

2.2 Attribute Predictor

The AP is of great importance to our model. The output $P(\mathbf{c}|S_v, S_c)$ is used by the AE to determine item relevance, and by the Attribute-aware Diversifying Decoder (ADD) (Sec. 2.3) to diversify the recommendations. The AP predicts $P(\mathbf{c}|S_v, S_c)$ with a small neural network. The input is the concatenation of the global preference: $[\mathbf{f}_v; \mathbf{f}_c]$. In doing so, the AP exploits features from both encoders when making the prediction. The AP consists of a hidden layer, with Batch Normalisation and ReLU activation, followed by the output layer activated with a sigmoid function. This prediction task is an auxiliary task, with an additional loss L_{AP} (see Section 2.4).

2.3 Attribute-aware Diversifying Decoder

The ADD is responsible for generating the diversified list of recommendations R_L. Inspired by IA-Select, a method used in web search [1], we incrementally extend R_L by selecting at each step the item with the highest score (Eq. 2), i.e., $v_i^* = \arg\max_{v_i} S(v_i)$. Specifically, at each step the ADD calculates S_{div} using $P(\mathbf{c} \mid S_v, S_c)$, obtained from the AP, as initial estimation of importance per category U:

$$S_{\text{div}}(v_i) = \sum_{j=1}^{|C|} U(c_j \mid S_v, S_c, R_L) \left(1 - V(v_i \mid c_j)\right), \tag{5}$$

where $V(v_i \mid c_j)$ represents the value of v_i in context of c_j, and is 1 if v_i belongs to category c_i and 0 else. After each step, we update $U(c_j \mid S_v, S_c, R_L)$ to reflect the newly added item to R_L by:

$$\begin{aligned} &U(c_j|S_v, S_c, R_L \cup \{v_i^*\}) = \\ &\quad \text{softmax}\left[\left(1 - V(v_i^* \mid c_j)\right) U(c_j|S_v, S_c, R_L)\right]. \end{aligned} \tag{6}$$

2.4 Losses

The loss L_{AE} calculated for the AE is the Cross-Entropy Loss.

$$\begin{aligned} L_{AE} &= \frac{-1}{|V|} \sum_{i=1}^{|V|} y_i \log P(v_i \mid S_v, S_c) \\ y_i &= 1 \text{ if } v_i = v_{T+1}, \text{ else } 0, \text{ where } v_{T+1} \text{ is the target.} \end{aligned} \tag{7}$$

Table 1: Dataset statistics after pre-processing.

	#Users	#Items	#Train seq.	#Valid seq.	#Test seq.	#Attributes
ML1M	6,041	3,261	784,309	93,929	97,871	18
TMall	31,855	58,344	698,081	54,706	54,705	71

To calculate L_{AP}, the Binary Cross-Entropy Loss is used, allowing for multi-labeled targets:

$$L_{AP} = \frac{-1}{|C|} \sum_{i=1}^{|C|} y_i \log P(c_i|S_v, S_c) + (1-y_i) \log (1 - P(c_i|S_v, S_c)) \quad (8)$$

$y_i = 1$ if $c_i = c_{T+1}$, else 0, where c_{T+1} is the target.

Then, to form the final loss used to optimize ADSR (L_{ADSR}), the two losses are interpolated with balancing parameter λ_{MT}:

$$L_{ADSR} = \lambda_{\text{MT}} \cdot L_{AE} + (1-\lambda_{\text{MT}}) \cdot L_{AP}. \quad (9)$$

3 EXPERIMENTAL SETUP

We seek to answer the following questions in our experiments: (RQ1) Where does the improvement of ADSR come from? What are the effects of the AE, AP and ADD modules? (See §4.1.) (RQ2) What is the performance of ADSR compared with non-attribute-aware diversity methods? (See §4.2.) (RQ3) How does the trade-off parameter λ_{s} affect the performance of ADSR? (See §4.3.)

3.1 Datasets

We evaluate ADSR on the following real-world datasets:

- MovieLens-1M,[1] which contains 1 million ratings from 6K users over 4K movies. Movies belong to at least one of 18 genres.
- TMall,[2] which is an e-commerce dataset containing 44.5M user interactions over 2.4M items belonging to one of 72 categories. We use the "buy" interactions only.

Genre and category are used as item attribute. Multi-labeled genres are handled by summing the embeddings.[3] Both datasets are filtered from users and items with less than 20 interactions. Per sequence, the first 80% is used as training set. The 20% is filtered from sequences with unseen items and halved to form the validation and test sets. The final inputs are sliding windows of size 10 [6] (10th item as target). Statistics after processing are reported in Table 1.

3.2 Evaluation metrics

Following [6–8], we measure accuracy of the model using *MRR@k* and *Recall@k*. Predictions made by the AP for multi-labeled targets are considered correct if one of the active labels is predicted. To measure diversity, we use *Intra-List Diversity (ILD)* [10]:

$$ILD(R_L) = \frac{2}{|R_L| \cdot (|R_L| - 1)} \sum_{i \in R_L} \sum_{j \neq i \in R_L} d(i,j), \quad (10)$$

where $d(i, j)$ is the euclidean distance between the one-hot-encoded item attributes of v_i and v_j. A second diversity measure is used where the number of unique attributes in the recommended list is counted, we refer to this as *discrete diversity* (*Dis*).

[1] https://grouplens.org/datasets/movielens/1m
[2] https://tianchi.aliyun.com/dataset/dataDetail?dataId=53
[3] In this work, a single genre/category is used as attribute, but ADSR can be modified to use other/multiple attributes (or contexts) using e.g. embedding summation.

3.3 Models for comparison

A number of SR methods have been proposed in the last few years. We do not compare with them because: (1) improving recommendation accuracy is out of the scope of this paper; and (2) many are not directly comparable due to the use of different model architectures. Therefore, we construct/select baselines that are fair (use the same information, similar architectures, etc.) to compare with:

- *Base Sequential Recommender (BSR)* uses only S_v to generate hidden representations. Then, BSR applies additive attention to get the global preference, used to generate R_L, similar to NARM [7].
- *Attribute-aware Neural Attentitive Model (ANAM)* incorporates attribute information and applies attention to the hidden representations, similar to ANAM [2]. We have adapted it to do SR rather than next-basket recommendation, and modified it to be similar to our own variants. The core principles remain the same, both apply attention to a unified representation based on S_v, S_c.
- *Multi Task Attribute-aware Sequential Recommender (MTASR)* extends ANAM to predict the attribute preference, and to use the weighted attribute embedding to predict R_L. This model can be regarded as a special case of ADSR with $\lambda_{\text{s}} = 1$.
- *ANAM+MMR* is ANAM with the Maximal Marginal Relevance reranking algorithm by Carbonell and Goldstein [3], trading off relevance and diversity by selecting v^* by:

$$v^* = \arg\max_{v_i \in V \setminus R_L} \lambda S(v_i) + (1-\lambda) \min_{v_j \in R_L} d(i,j), \quad (11)$$

where V are all items not in R_L and $d(i, j)$ is the same as in Eq. 10. This model does not have the AP module, so it cannot do attribute-aware diversification based on preference.

Besides ANAM+Maximal Marginal Relevance (MMR), none of the methods listed considers diversity when recommending items. ADSR learns attribute preferences to provide an attribute-aware diversified list of recommendations.

3.4 Implementation details

For a fair comparison, we use the same settings for all models. We set $d_v = d_c = d_{\text{GRU}} = 128$. Embeddings are initialized using the Xavier method. Dropout is applied to the embeddings and hidden representations with $p = 0.5$. We use the Adam optimizer with learning rate 0.01, batches of 1024 samples for 60 epochs. $\mathbf{f}_v$ is projected to d_v by one linear layer in the case of *BSR* and *ANAM*. A bilinear mapping is used to combine and project $\mathbf{f}_v$ and $\mathbf{f}_c$ in *MTASR* and *ADSR*. The best model is selected by MRR and Recall on the validation set.[4]

4 EXPERIMENTAL RESULTS

4.1 Ablation study

To answer (RQ1) and to show the effectiveness of the AE, AP and ADD modules, we compare the results of variations of ADSR, as shown in Table 2. BSR is ADSR without all three modules (no attribute information is used). Further, ANAM is ADSR without AP and ADD, and MTASR is ADSR without ADD.

First, the AE is effective. This is demonstrated by the fact that ANAM outperforms BSR on both datasets, showing that modeling attribute information significantly improves performance. Second,

[4] The code from this paper is available at https://github.com/antonsteenvoorden/ADSR.

Table 2: Performance of recommendation models. The best performing model is boldfaced, the second best is underlined. Statistical significance is determined by a two-tailed t-test. Pairwise differences of all models vs. *BSR* are significant with $p \leq .01$. For *ADSR* vs. *ANAM+MMR* significance with $p \leq .05$ is indicated by $^{\Delta}$, the other results are significant with $p \leq .01$.

	ML1M ANAM+MMR: λ_s@10 = 0.2, λ_s@20 = 0.3 ADSR: λ_s@10 = 0.25, λ_s@20 = 0.4, λ_{MT} = 0.9									**TMall** ANAM+MMR:λ_s@10 = 0.01, λ_s@20 = 0.01 ADSR: λ_s@10 = 0.01, λ_s@20 = 0.01, λ_{MT} = 0.4								
Model	MRR		Recall		ILD		Dis.		AP Acc.	MRR		Recall		ILD		Dis.		AP Acc.
	@10	@20	@10	@20	@10	@20	@10	@20		@10	@20	@10	@20	@10	@20	@10	@20	
BSR	0.0728	0.0793	0.0728	0.2805	1.4197	1.452	7.7593	10.2603	–	0.0863	0.0891	0.1601	0.2013	0.9141	0.9428	5.0294	7.9399	–
ANAM	0.0777	0.0840	0.1944	0.2871	1.4283	1.4601	7.8751	10.4070	–	0.0983	0.1015	0.1835	0.2298	0.8374	0.8725	4.6215	7.3883	–
MTASR	**0.0804**	**0.0868**	**0.2000**	**0.2936**	1.4250	1.4512	7.9025	10.4311	0.5292	**0.0984**	**0.1018**	**0.1848**	**0.2332**	0.7680	0.8069	4.2306	6.7749	0.3820
ANAM+MMR	0.0772	0.0839	0.1920	0.2852	1.5367	1.5016	9.9375	11.9467	–	0.0978	0.1007	0.1808	0.2235	0.9417	1.0095	5.9315	11.8666	–
ADSR	0.0795	0.0862	0.1939	0.2859	**1.6222**	**1.5025**	**10.7988**	**12.7599**	0.5292	0.0963$^{\Delta}$	0.0989	0.1744	0.2108	**1.0267**	**1.1275**	**6.5234**	**13.8204**	0.3820

the AP further enhances performance of the model, as MTASR outperforms ANAM. This shows that learning attribute preference with the AP brings a significant improvement to the model. Third, the ADD significantly improves diversity without hurting accuracy. This claim is supported by the fact that ADSR outperforms MTASR on both diversity metrics. MTASR outperforms ADSR in terms of MRR and Recall, which is expected as diversity is increased by trading off relevance, where MTASR is equal to ADSR with λ_s=1. However, the trade-off is advantageous, as gains in diversity are made of 39% and 14% over BSR in ILD@10 and Dis@10, with a reduction of merely 1.24% in MRR@10 and 3.29% in Recall@10.

4.2 Comparison with non-attribute-aware diversity methods

To answer (RQ2), we compare ADSR with ANAM+MMR. By incorporating MMR into ANAM, ANAM+MMR is able to provide diversified recommendations. From Table 2 we see that ADSR is able to provide more diverse recommendations on both datasets. On ML1M, ADSR significantly outperforms ANAM+MMR on all metrics, while increasing diversity: ADSR is more effective at diversifying results, with an increase of 8.67% and 5.56% over ANAM+MMR in ILD@10 and Dis@10, while yielding higher MRR and Recall. On TMall, ADSR is best at diversifying results, showing increases of 9.98% and 9.03% over ANAM+MMR in terms of Dis@10 and ILD@10, and gains of 16.46% and 11.69% in Dis@20 and ILD@20. However, ADSR yields lower accuracy than ANAM and ANAM+MMR, which is likely due to the high sparsity of TMall and the larger attribute space, making it harder for the AP to correctly predict attributes.

4.3 Effect of the trade-off hyperparameter λ_s

To answer (RQ3), we vary λ_s from 0.0 to 1.0 and investigate the effect it has on accuracy and diversity (for brevity, we chose a single metric of each type; see Figure 2). When λ_s=0, it maximizes diversity only, and does this successfully, yielding the maximum diversity possible on both datasets. On TMall, very low values of λ_s are required to diversify results as the item space is much larger, resulting in very small values for S_{div}. Clearly, S_{div} only affects item selection when it is weighed much heavier than S_{rel}. Finally, Figure 2 shows that diversity decays slower than accuracy increases, meaning that ADSR can effectively achieve a large gain in diversity for a small loss of accuracy.

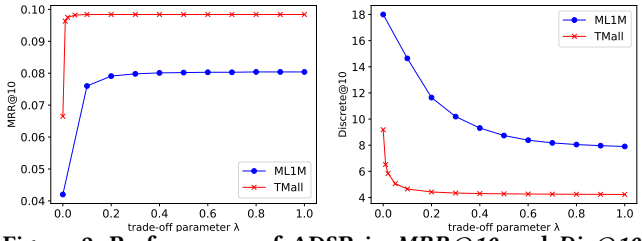

Figure 2: Performance of ADSR in *MRR@10* and *Dis@10*, while varying λ_s (in Eq. 2) from 0.0 to 1.0 on two datasets.

5 CONCLUSION AND FUTURE WORK

We propose a novel Attribute-aware Diversifying Sequential Recommender that utilizes attribute information when modeling a user's sequential behavior to simultaneously learn their most likely item to interact with and their preference for attributes. Experimental results on two datasets show that ADSR effectively diversifies the list of recommendations based on the predicted preference for attributes, trading in a controllable and small amount of accuracy for large gains in diversity, through hyperparameter λ_s. Further, ablation shows the importance and the positive effect of incorporating attribute information and attribute prediction.

A limitation of ADSR is that performance decreased on TMall, due to sparsity and a larger item and attribute space. However, dealing with sparsity is a common issue in SR. As to future work, we would like to address the limitation of the AP by looking into ways of leveraging auxiliary information such as user profiles, item reviews, etc. as we think high AP accuracy increases performance.

REFERENCES

[1] Agrawal et al. 2009. Diversifying Search Results. In *WSDM*. 5–14.
[2] Bai et al. 2018. An Attribute-aware Neural Attentive Model for Next Basket Recommendation. In *SIGIR*. 1201–1204.
[3] Carbonell and Goldstein. 1998. The Use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. In *SIGIR*. 335–336.
[4] Rich Caruana. 1997. Multitask Learning. *Machine Learning* 28, 1 (1997), 41–75.
[5] Chen et al. 2019. AIR: Attentional Intention-aware Recommender Systems. In *ICDE*. 304–315.
[6] Hidasi et al. 2016. Session-based Recommendations with Recurrent Neural Networks. In *4th International Conference on Learning Representations, ICLR 2016*.
[7] Li et al. 2017. Neural Attentive Session-based Recommendation. In *CIKM*. 1419–1428.
[8] Ren et al. 2019. RepeatNet: A Repeat Aware Neural Recommendation Machine for Session-based Recommendation. In *AAAI*.
[9] Xiaoyuan Su and Taghi M Khoshgoftaar. 2009. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009).
[10] Mi Zhang and Neil Hurley. 2008. Avoiding Monotony: Improving the Diversity of Recommendation Lists. In *RecSys*. 123–130.