Adaptive Hierarchical Translation-based Sequential Recommendation

Yin Zhang
Texas A&M University
zhan13679@tamu.edu

Yun He
Texas A&M University
yunhe@tamu.edu

Jianling Wang
Texas A&M University
jlwang@tamu.edu

James Caverlee
Texas A&M University
caverlee@cse.tamu.edu

ABSTRACT

We propose an adaptive hierarchical translation-based sequential recommendation called HierTrans that first extends traditional item-level relations to the category-level, to help capture *dynamic* sequence patterns that can *generalize across users and time*. Then unlike item-level based methods, we build a novel hierarchical ***temporal graph*** that contains item multi-relations at the category-level and user dynamic sequences at the item-level. Based on the graph, HierTrans *adaptively* aggregates the high-order multi-relations among items and dynamic user preferences to capture the dynamic joint influence for next-item recommendation. Specifically, the user translation vector in HierTrans can *adaptively change* based on both a user's previous interacted items and the item relations inside the user's sequences, as well as the user's personal dynamic preference. Experiments on public datasets demonstrate the proposed model HierTrans consistently outperforms state-of-the-art sequential recommendation methods.

ACM Reference Format:
Yin Zhang, Yun He, Jianling Wang, and James Caverlee. 2020. Adaptive Hierarchical Translation-based Sequential Recommendation. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3366423.3380067

1 INTRODUCTION

Sequential recommendation aims to recommend new items based on a user's recent behaviors, e.g., to recommend a smart home device after a user purchases a smart home hub [6, 13]. Existing sequential recommenders mainly focus on modeling sequential patterns by using user activity sequences, such as Markov Chains [8], Recurrent Neural Networks, and Convolutional Neural Networks [11, 27]. However, purely sequence-based recommendation usually faces challenges in capturing *general item relations* that are not easily discovered from highly-personalized user sequences. For example, Figure 1 shows how a purely sequence-based recommender will treat the two user sequences as fundamentally different, even though there are clear patterns among the kinds of items being purchased (in this case, laptops and accessories). Though the specific

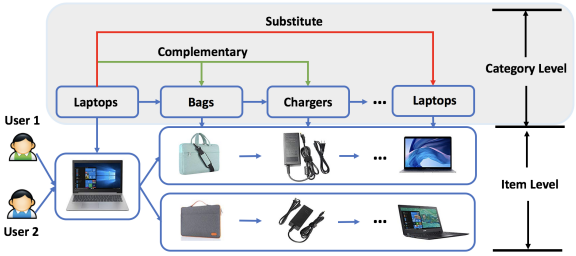

Figure 1: The hierarchical structure for two user sequences. Though the two users purchase different items in sequences, they have the same sequence patterns at the category-level and with respect to category-level item relations.

items are different in each sequence, some items are complements to each other, while others are substitutes. Hence, there is growing interest in capturing these kinds of multi-relations for improved recommendation [14, 28, 33] and in particular, of leveraging complementary and substitute relations for their important influence on user purchases [24, 30, 39].

While encouraging, such relation-aware sequential recommendation still faces several key challenges: (i) *Sparsity and Temporal Generalization:* Previous works mainly use item-level relations to improve non-time aware user recommendation. However, both item relations and user sequences are typically very sparse since users interact with very few items. More importantly, these item-level relations become less useful as items are updated over time (e.g., an older iPad being replaced by a new model). Thus, we explore how to view item relations at the category-level as well, since these categorical relations are denser and more stable over time; (ii) *Hierarchical Structure:* While most previous methods can directly connect item-level relations with user interactions (since sequences are viewed from an item perspective), a categorical-level perspective introduces a hierarchical structure of category-level item relations and user-based item-level sequences. Hence, an important question is how to organize the hierarchical connections so that we can extract the complex multi-relations among items that are revealed inside user dynamic sequences; (iii) *Personalized Dynamic Adaptation:* Translation-based recommendation has received lots of attention for strong performance with high scalability to large, real-world datasets [13, 23]. These methods treat users as translation vectors to connect items in a translation space, a natural fit for capturing the interaction between users and the relations among items. However, most translation-based methods model user translation behavior identically. In practice, user translation behavior can be influenced

WWW '20, April 20–24, 2020, Taipei, Taiwan

ACM ISBN 978-1-4503-7023-3/20/04.
https://doi.org/10.1145/3366423.3380067

not only by a user's previous interacted items and the item relations among those items, but also the user's personal preference towards items and item relations. Thus, this interplay is crucial for relation-aware sequential recommendation.

Considering these challenges, we propose a hierarchical translation-based recommendation method called HierTrans. HierTrans has three unique properties: **First**, HierTrans extends traditional item-level relations to the category-level, to help capture dynamic sequence patterns that can generalize across users and across time. Furthermore, these category-level item relations can effectively alleviate the sparsity problem in both item relations and user sequences; **Second**, HierTrans is built on a hierarchical temporal graph $\mathcal{G}$ that contains item multi-relations at the category-level and user dynamic sequences at the item-level. The hierarchical graph structure enables us to more easily extract the high-order complex relation patterns among items that are revealed inside user dynamic sequences; **Third**, based on $\mathcal{G}$, we propose a novel hierarchical translation-based recommendation method that adaptively aggregates item multi-relations at the category-level and dynamic user preferences at the item-level for next-item recommendation. Specifically, the user translation vector in HierTrans can *adaptively change* based on both a user's previous interacted items and the item relations inside the user's sequences, as well as the user's personal dynamic preference.

To the best of our knowledge, this work is one of the first to aggregate category-level item relations to dynamically adapt user translation behavior in translation-based recommendation. Through extensive experiments on three public datasets, HierTrans consistently outperforms state-of-the-art methods by 7.02% in recall and 7.72% in NDCG (on average against the next-best alternative). We also evaluate different components of HierTrans to better understand their impact on sequential recommendation.

2 RELATED WORK

Sequential Recommendation: Sequential dynamics play a key role in many modern recommenders [1, 13, 36, 37]. Typically, there are two major types of sequence models: (i) the order-based models (shown in Figure 2 (a)) consider user sequences as *item orders* and focus on uncovering diverse patterns from these orders, such as using Markov Chains [8, 26], Recurrent Neural Networks (RNNs) [5, 11, 12], Convolutional Neural Networks (CNNs) [27, 36, 37] and attention mechanism [13]; (ii) the translation-based models (shown in Figure 2 (b)) treat user sequences as *user translation behavior* to connect items [6, 7, 23]. These translation-based methods can capture higher-order user-item interactions [6] and are more scalable compared with neural network-based models. However, most assume each user translation vector is static and identical, thus the translation behavior is the same across time; (iii) Our proposed model, HierTrans, as shown in Figure 2 (c), adaptively aggregates item high-order multi-relations at the category-level and dynamic user preferences at the item-level for next-item recommendation. Thus the translation vector can adaptively change.

Item Relations in User Sequences: Different types of item relations have gained attention to improve recommendation [4, 14, 33]. Among these, complementary and substitute relations [20, 21, 30, 39] highly influence user *dynamic* sequences. Recently, many works focus on those item relations due to their pervasive real-world application, such as inferring complementary and substitute relations based on content information since these relations are very sparse [24, 30, 39]. Here, we mainly focus on the category-level relations. Furthermore, few of these methods explore such complement and substitute relations for *dynamic* sequential recommendation.

Table 1: Notation.

Notation	Explanation
$\mathcal{U}, \mathcal{I}, C$	user set, item set, category set
$\mathcal{S}^u$	historical sequence for user u, $\{S_1^u, S_2^u, \dots S_{\lvert S^u \rvert}^u\}$
$S_{:n,T}^u$	the user u previous T interacted items $S_{n-T+1}^u \dots S_n^u$
$\mathcal{G}$	the hierarchical graph, contains $\mathcal{G}^C$ and $\mathcal{G}^I$
$\mathcal{G}^C$	category-level item relation graph
$c^i, \vec{c}^i$	item i category and the category embedding vector
(c^i, r_k, c^j)	category of item i and category of item j are connected by the r_k relation
$\mathcal{G}^I$	item-level user sequence graph
$i, \vec{i}$	the item i and the item embedding vector
(i, r_u, j)	item i and j are connected by the r_u relation

Translation-based Method in Graph: Many research efforts have investigated graph structures [16] for link prediction [2, 19, 22, 28, 29, 35, 38]. Specifically, for translation-based models [17, 31], TransE [3] first proposed the core idea that items were connected by translation vectors in their vector space. The model structure is simple but achieves powerful performance in many situations. In follow-up work, various methods (such as TransH [32] and TransR [18]) extend TransE. Different from these methods that are mainly based on *generic* graphs, we investigate the specific structure of a user's dynamic sequence (which is a path) inside an item's heterogeneous relational graph.

3 PROPOSED METHOD: *HierTrans*

We aim to provide a personalized sequential recommendation that takes advantage of multi-relations between items.

Problem Statement. Formally, we assume a set of users $\mathcal{U}$, items $\mathcal{I}$ and categories C where $c^i \in C$ denotes the category of item i. For each user, we have a sequence of items $\mathcal{S}^u = \{S_1^u, \dots S_{\lvert S^u \rvert}^u\}$ that u has interacted with. Besides user-item interaction sequences, we assume there are also multi-types of item relations r_k in relation set $\mathcal{R}$. Here we focus on complementary r_c and substitutes r_s relations. A triple (i, r_k, j) denotes there exists a type of relation $r_k \in \mathcal{R}$ between item i and j from $\mathcal{I}$. Our task is: given a user sequence $\mathcal{S}^u = \{S_1^u, \dots S_{\lvert S^u \rvert}^u\}$, we seek to predict the next item for the user.[1]

Based on the task, we face two key questions: (1) First, how can we *organize* the item multi-relations with user sequences to facilitate modeling the dynamic user sequential behavior? (2) Second, user behavior can change based on both the relations of previously interacted items and the user's dynamic personalized preference. How can we *model* the joint influence of user preference and item multi-relations, that at the same time, can also *adaptively adjust* to

[1]Notations are shown in Table 1. The category of an item is denoted by c with upper corner of the item letter (e.g., item i's category is c^i). The vector embedding of items and relations are denoted by the same letters with $\vec{\cdot}$. That is, the vector of item i is denoted as $\vec{i}$ and the relation r_k embedding vector is denoted as $\vec{r}_k$. Matrices are represented by boldface uppercase characters.

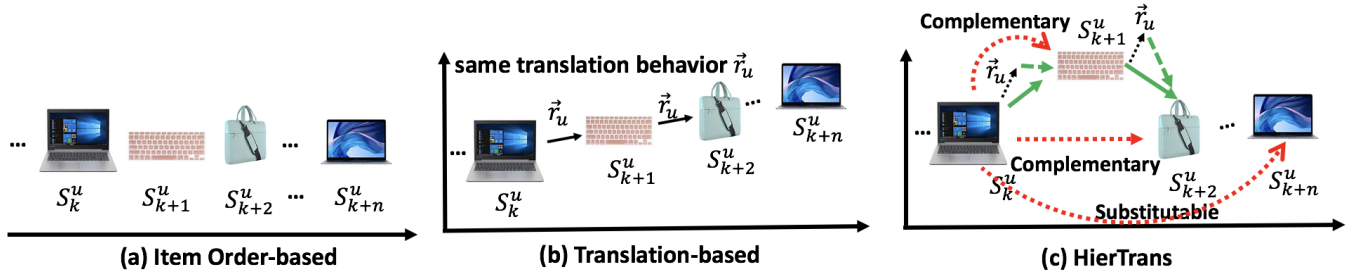

Figure 2: Different sequence models. S_k^u denotes that user u interacted with items in sequence order. $\vec{r}_u$ is the user translation vector. (a) Item order-based models, such as RNN, CNN, and Markov Chains, mainly focus on the diverse patterns of item orders in user sequences; (b) Translation-based sequence models treat users as translation vectors to connect items, which models 'higher-order' interactions between a user, her previously interacted items and the next interacted item. Since $\vec{r}_u$ stays the same in the user sequence, these models assume a user's translation behavior connecting items is the same across time; (c) Our proposed HierTrans considers both user personal preferences and item relations. The translation behavior (green line) can adaptively change according to both user preference and the relations of her recent interacted items, making the model more flexible to capture user complex dynamic preferences over time.

the user's dynamic personal preferred item/item relations? We deal with each of these questions in the following.

3.1 Hierarchical Temporal Graph

We organize/build a hierarchical temporal graph $\mathcal{G}$ to facilitate modeling the dynamic joint influence of the item relations and user personalization. Specifically, the graph $\mathcal{G}$ contains three parts, as shown in Figure 3(a)(b): the graph $\mathcal{G}^C$ captures item multi-relations at the category-level; the graph $\mathcal{G}^I$ captures each user's dynamic sequence at the item-level; and finally, the connections between the two graphs to integrate the hierarchical connections. In the following, we detail each step of this construction in order.

Item Multi-Relations Graph: The first graph $\mathcal{G}^C = (V^C, E^C)$ is built by the category-level item connections to facilitate exploiting item high-order semantic multi-relations. The nodes are item categories $V^C = \cup_{i \in I} c^i$. Based on [30], we extend the item relations into category-level relations with the following rule: if item i complements/substitutes item j, then the category c^i of item i complements/substitutes category c^j of item j. That is [30]:

- For complementary: if $(i, r_c, j) \Rightarrow (c^i, r_c, c^j) \in \mathcal{G}^C$;
- For substitutable: if $(i, r_s, j) \Rightarrow (c^i, r_s, c^j) \in \mathcal{G}^C$;

We use the category-level relations since the choices of specific items are highly personalized in user sequences and item-level relations in existing datasets are extremely sparse [21, 39]. We observe that, category-level relations [30] are highly relevant to user sequences, and can usually provide denser and generic item relation information that can be applied for different user sequences. Furthermore, for sequential recommendation, we should ensure the model can be *generalized with time drift*. Considering that items can be updated over time, the category-level relations are more stable in time dimension. For example, an iPad2 may be updated to iPad3, but they both belong to the same tablet category. Thus the category-level relations of the iPad2 can also be considered to the iPad3 [30], such as being complementary to the earbud category.

Therefore, we extend the item-level relations to category-level based on [30] in order to capture category-level *semantic information* to help sequential recommendation. That it, we want to capture if item i and j are related, their categories probably share similar semantic information under r_k, and their representations are closer connected by r_k [30]. For example, if a laptop and mouse are complementary, we can infer the categories of the laptop and mouse are closer correlated in complementary relations. Furthermore, relations among different items also contain rich information. For example, if we know both a keyboard and mouse are complementary to a laptop, then keyboard and mouse are closer correlated. The built graph structure smooths the way to investigate them.

Dynamic User Interactions Graph: The second graph $\mathcal{G}^I = (V^I, E^I)$ is built by user sequences to facilitate exploiting user dynamic preference towards specific items. The nodes in $\mathcal{G}^I$ are specific items $V^I = \cup_{i \in I} i$. Concretely, we treat user sequences as a series of transitions the user u has made between each two adjacent items in S^u. Each user represents one type of relation as r_u. In sum,

- For user $u \in \mathcal{U}$, if item j is next to item i in the user sequence S^u, then $(i, r_u, j) \in \mathcal{G}^I$;

Thus, the node connections in $\mathcal{G}^I$ can dynamically change with user sequences changes.

Connecting the Two Graphs: Last, we connect $\mathcal{G}^C$ and $\mathcal{G}^I$:

- For item i, if item i belongs to category c^i, then we connect them by the *belongs to* relation: (i, r_b, c^i);

where r_b represents the *belongs to* relation. The connections naturally aggregate the complex influence from both item category-level relations and user sequences as shown in Figure 3(b).

3.2 Recommendation with HierTrans

This section introduces HierTrans that explores the dynamic joint influence between user personal preference and item multi-relations for next item prediction based on the graph $\mathcal{G}$, as shown in Figure 3(c)(d). Since translation-based methods have shown success at capturing user-item interactions [6], it motivates us to utilize its structure to investigate the user-item relation interactions in dynamic sequences. The basic idea of traditional translation-based recommendation is: user sequences, e.g., $S^u = \{S_1^u, S_2^u ... S_{|S^u|}^u\}$, are composed by triples <head, translation relation, tail>, where the head represents the item that user has previously interacted with, and the tail is the next item. They satisfy: $\vec{S}_n^u + \vec{r}_u \approx \vec{S}_{n+1}^u$.

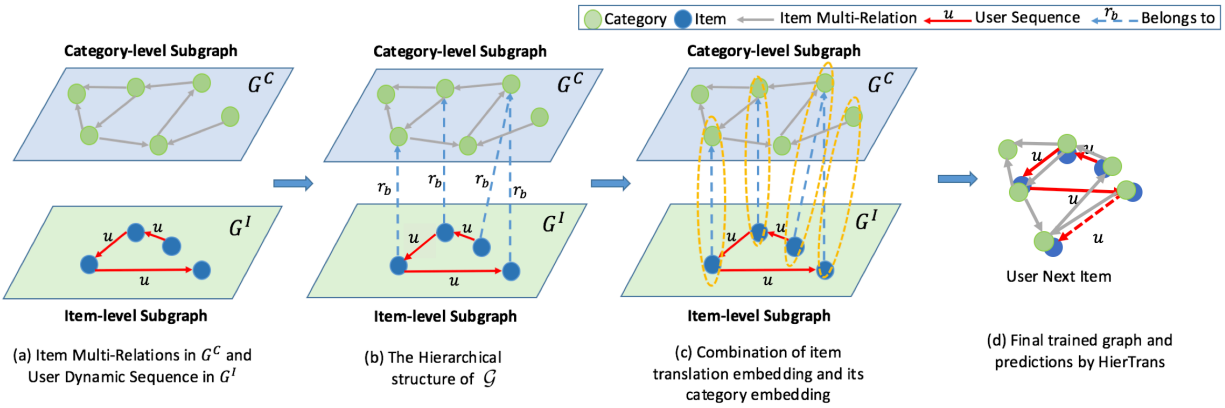

Figure 3: Overview of HierTrans on the hierarchical temporal graph $\mathcal{G}$. Dots in $\mathcal{G}^C$/$\mathcal{G}^I$ represent categories/items. Category nodes are connected (grey lines) by item relations in $\mathcal{G}^C$. Each user connects item nodes by their sequences (red lines) in $\mathcal{G}^I$. The item-category connections (in blue dotted lines) represent the "belongs to" relations shown in (c). With HierTrans, we can predict a user's next item based on both user preference and the item relations inside the user sequence, as shown in (d).

HierTrans adaptively aggregates the embedding based on $\mathcal{G}^C$ and $\mathcal{G}^I$. The high-level idea of HierTrans can be formulated as:

$$\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T}) + \vec{Trans}(r_u | S^u_{:n,T}, r_k) \approx \vec{S}^{*u}_{n+1}, \quad (1)$$

where $S^u_{:n,T} = [S^u_{n-T+1} ... S^u_n]$ is the user u's previous T interacted items and S^u_{n+1} is the next item. For each term in Equation (1): (i) The head $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$ captures both user personal dynamic preferred patterns through $\mathcal{G}^I$ and item multi-relation patterns through $\mathcal{G}^C$ inside the user's recent sequences $S^u_{:n,T}$ by function $\vec{Head}$; (ii) We propose a novel user translation vector $\vec{Trans}(r_u | S^u_{:n,T}, r_k)$, which models a user's dynamic personal preferred item relations. Different from previous translation-based recommendation where the user translation vector $\vec{r}_u$ is static and identical across time, this proposed user translation vector can *adaptively change* based on both a user's previous interacted items and item relations r_k inside the user's sequences, as well as the user's personal preference. (iii) Similar to translation-based methods, the tail is the embedding of user next item learned by HierTrans. In the following, we show the detailed construction.

3.2.1 *Construction of Relation-aware Head* $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$.

The head of HierTrans contains both item relations from $\mathcal{G}^C$ and user personal preference from $\mathcal{G}^I$.

I. Item Category-level Multi-relation in $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$. To extract item relations inside user sequences, we first learn the item category relation-aware embeddings in $\mathcal{G}^C$. TransE is leveraged (note that TransH and TransR can also be easily applied):

$$\vec{c}^i + \vec{r}_k \approx \vec{c}^j, \quad k \in \{c, s\}. \quad (2)$$

This means that when (c^i, r_k, c^j) holds, the item j category embedding $\vec{c}^j$ should be the nearest neighbor of $\vec{c}^i + \vec{r}_k$.

II. User Personal Preference in $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$. In $\mathcal{G}^I$, different from item relations which *pair* each two nodes, items in user sequences are connected in *order*. Thus, we explore the *unified items* effects. That it, we consider the recent T items $S^u_{:n,T} = [S^u_{n-T+1} ... S^u_n]$ together to extract the user preference. Based on $S^u_{:n,T}$, we can follow recent models to extract the different patterns (such as the union and skip patterns by CNN [27]) in $S^u_{:n,T}$. Thus, $\vec{Head}$ here is a function which returns a vector that has the same dimension as the user translation vector, such as the attention and CNN. We discussed the choice of $\vec{Head}$ in Section 4.2.

III. Relation-aware Pattern Learned by $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$. With the item embedding $\vec{i}$ based on $\mathcal{G}^I$ and item category-level relation-aware embedding $\vec{c}^i$ based on $\mathcal{G}^C$, how can we incorporate them to capture the dynamic item relations *inside* user sequences? Different from other relations, the item and its category are very closely related to each other. So we connect $\mathcal{G}^I$ and $\mathcal{G}^C$ by adding the corresponding embedding vectors: if item i's category is c^i, then

$$\vec{i}^* = \vec{i} + \vec{c}^i, \quad (3)$$

as shown in Figure 3(c)(d). That is, for each item in $S^u_{:n,T}$, we apply Equation (3) to construct $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$. Thus HierTrans considers the previous T items relation-aware patterns that contain both dynamic user preference and item multi-relations.

3.2.2 *Construction of Adaptive Translation* $\vec{Trans}(r_u | S^u_{:n,T}, r_k)$.

Many existing translation-based recommendations assume user translation vector is static and identical [6, 23]. However, in practice, user translation behaviors are dynamic based on both (1) previously interacted items and those item relations; (2) personal preference towards specific items and personal preference towards item relations. For example, some users frequently change cellphones while others do not. For those users, substitutable relation is frequently utilized. Considering that, we propose a *novel adaptive translation vector* that considers these influence factors.

We first construct the candidates of user translation choice based on user preference and item relations:

$$\vec{r}_{uk} := \vec{t} + \vec{t}_u + \vec{r}_k, \quad k \in \{c, s, n\}, \quad (4)$$

where $\vec{t}$ denotes the global transition dynamics across all users [6]. $\vec{t}_u$ represents user personal preference translation. $\vec{r}_k$ represents embedding of different item relations ($\vec{r}_n$ is "not related").

Table 2: Amazon datasets. Sparsity is item sparsity in the user-item matrix.

Dataset	Users	Items	Feedback	Categories	Relations	Item Sparsity
Electronics	11,965	22,791	303,125	622	174,255	0.111%
C & A	23,539	11,170	173,464	50	4,704	0.066%
H & K	19,231	20,098	251,825	853	222,911	0.065%

Then we use the attention mechanisms [10, 34] to capture the relation that the user would choose for the next item:

$$\begin{aligned} \vec{h}_{uk} &= \vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T}) + \vec{t}_u + \vec{r}_k \\ \vec{u}_{uk} &= tanh(\mathbf{W}_a \vec{h}_{uk} + \vec{b}_a) \\ \alpha_{uk} &= \frac{exp(\vec{u}_{uk}^\top \vec{u}_w)}{\sum_k exp(\vec{u}_{uk}^\top \vec{u}_w)} \\ \vec{Trans}(r_u | S^u_{:n,T}, r_k) &= \sum_k \alpha_{uk} \vec{r}_{uk}, \end{aligned} \tag{5}$$

where $\mathbf{W}_a$, $\vec{b}_a$ and $\vec{u}_w$ are learnable [34]. Based on Equation (5), through attention mechanisms and the construction of $\vec{r}_{uk}$, user's translation behaviors are *adaptively changed* according to user previous interacted items and their relation patterns (by $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T})$), her personal preference (by $\vec{t}_u$) and her preferred item relations (by $\vec{r}_k$). Hence, with different previous interacted items, the translation vector is adaptively changed.

3.3 Optimization

The loss function of HierTrans contains the user dynamic personal preference learning, item multi-relations learning and regularizer:

$$\arg\min_{\Omega} \mathcal{L}^I(\mathcal{G}^I) + \mathcal{L}^C(\mathcal{G}^C) + R(\Omega).$$

$\mathcal{L}^I(\mathcal{G}^I)$ is user item-level personal preference learning and $\mathcal{L}^C(\mathcal{G}^C)$ is item category-level relation learning. Ω is the set of model parameters. $R(\Omega)$ is L_2 regularizer here [6]. For $\mathcal{L}^I(\mathcal{G}^I)$ [6]:

$$\mathcal{L}^I(\mathcal{G}^I) = -\sum_{u \in \mathcal{U}} \sum_{j \in \mathcal{S}^u} \sum_{j' \notin \mathcal{S}^u} ln\, \sigma(\hat{p}_{u, S^u_{:n,T}, j} - \hat{p}_{u, S^u_{:n,T}, j'}),$$

where j denotes the next item of S^u_n in user sequence S^u. $\sigma(\cdot)$ is the sigmoid function. $\hat{p}_{u, S^u_{:n,T}, j}$ means the probability that u will prefer j given previous T items, which is calculated by:

$$\hat{p}_{u, S^u_{:n,T}, j} \propto \beta_j - d(\vec{Head}(\mathcal{G}^I, \mathcal{G}^C | S^u_{:n,T}) + \vec{Trans}(r_u | S^u_{:n,T}, r_k), \vec{j}^*), \tag{6}$$

where β_j is item bias. We take $||\vec{x} - \vec{y}||^2$ as the dissimilarity measure $d(\vec{x}, \vec{y})$ to calculate the probability. For $\mathcal{L}^C(\mathcal{G}^C)$ [3]:

$$\begin{aligned} \mathcal{L}^C(\mathcal{G}^C) = \sum_{k \in \{c,s\}} \sum_{(c^i, r_k, c^j) \in \mathcal{R}} \sum_{(c^{i'}, r_k, c^{j'}) \in \mathcal{R}'} max(0, \gamma \\ + d(\vec{c}^i + \vec{r}_k, \vec{c}^j) - d(\vec{c}^{i'} + \vec{r}_k, \vec{c}^{j'})), \end{aligned}$$

where $\mathcal{R}'$ represents the negative sets $\mathcal{R}' = \cup_{k \in \{c,s\}}(\{(i', r_k, j) \cup (i, r_k, j')\})$. (i', r_k, j) means $i', j \in \mathcal{I}$ but they are not related by r_k. Here $d(\cdot, \cdot)$ uses the same measurement as Equation (6). The $\gamma > 0$ is a margin hyperparameter [3].

4 EXPERIMENTS

We adopt three public datasets from Amazon [21]: Electronics (Elec), Cell Phones & Accessories (C & A), and Home & Kitchen (H & K), as shown in Table 2. They contain rich types of item relations and user purchase sequences. Following prior work [13, 27], we convert all numeric ratings to implicit feedback of 1. We discard users having less than n feedbacks (n is 20 for Elec, 5 for C & A, and 5 for H & K to obtain different sparsity and number of item relations). We also remove items having less than 3 feedbacks to keep the original user sequence patterns and alleviate the cold-start problem. Category-level relations are trained based on the item-level relation frequency to alleviate noise.

Similar to prior work in this area [13], we split the user sequences S^u into three parts: the most recent interacted item in each user sequence ($S^u_{|S^u|}$) for testing; the second most recent interacted item ($S^u_{|S^u|-1}$) as the validation data; the remaining items are used as training data.

Evaluation Metrics. To be consistent with prior work in sequential recommendation [13, 27], we adopt two common top-k metrics – recall@k and NDCG@k [13] – for evaluating recommendation performance. The recall at top-k measures the fraction of user next items that have been predicted over all purchased items. NDCG@k considers the position of correctly recommended items. Following the setup used in prior work [9, 13, 15], we randomly sample 100 negative items for each user and rank these items with the ground-truth items to avoid heavy computation on all user-item pairs. The evaluation metrics can be calculated based on the 101st items.

Baselines. We compare HierTrans with the following baselines:

- *BPR* [25]. This is the standard Bayesian personalized ranking (BPR) framework using matrix factorization;
- *TransE* [3]. We use user sequences to build the graph and recommend items based on a nearest neighbor search of the item embeddings. Notice other translation based methods, e.g., TransH and TransR, can be easily adapted for HierTrans, thus here we focus on TransE as a representative method;
- *TransFM* [23]. For the recent TransFM, we consider the item's category relation as side information;
- *TransRec* [6]. It treats each user as a translation vector to connect items by user sequences and learn item embeddings;
- *GRU4Rec* [11]. This is a session-based recommender based on RNN. We treat each user's sequence as a session;
- *Caser* [27]. A state-of-the-art sequential recommendation method based on a CNN to deal with high-order Markov Chains;
- *SASRec* [13]. The recent state-of-the-art sequential recommendation method uses a self-attention mechanism to capture useful user sequence patterns. We also use two self-attention blocks;

Parameter settings. The number of latent dimensions is empirically set to be 200 and attention dimension is 100 for all methods. $T = 10$. $\gamma = 1.0$. The regularizer is chosen by grid search from {0.5, 0.1, 0.05, 0.01, ... 0.00001}, drop out rate is from {0.1, 0.3, 0.5, 0.7, 0.9} and the learning rate is from {0.0001, 0.001, 0.01, 0.1}. The negative sampling ratio is 1. For HierTrans, other hyperparameters are tuned based on the validation data. $\vec{Head}(\cdot)$ is the average function for the three datasets. We discussed the choices of $\vec{Head}(\cdot)$ (such as attention and convolution) in Section 4.2. *All experimental settings, data, and code can be found at http://people.tamu.edu/~zhan13679/.*

Table 3: Evaluating HierTrans versus baselines over three datasets (for all metrics, higher is better). The percentage improvement compares HierTrans versus the next-best alternative.

Dataset	Metric	BPR	TransE	TransFM	GRU4Rec	Caser	TransRec	SASRec	HierTrans	Improv.
Elec	R@1	0.1073	0.1161	0.1296	0.1153	0.1570	0.1678	0.1799	**0.1927**	7.1%
	R@5	0.2800	0.2828	0.3200	0.2959	0.3830	0.4028	0.3954	**0.4551**	13.0%
	R@10	0.3942	0.3941	0.4410	0.4162	0.4997	0.5313	0.5117	**0.5891**	10.9%
	N@5	0.1996	0.2020	0.2267	0.2029	0.2739	0.2892	0.2925	**0.3285**	12.3%
	N@10	0.2363	0.2390	0.2653	0.2435	0.3117	0.3308	0.3303	**0.3721**	12.5%
C&A	R@1	0.1359	0.1197	0.1328	0.1393	0.1423	0.2153	0.1848	**0.2300**	6.8%
	R@5	0.3145	0.2974	0.3295	0.3455	0.3615	0.4660	0.4146	**0.4775**	2.5%
	R@10	0.4144	0.4006	0.4419	0.4742	0.4397	0.5925	0.5327	**0.6031**	1.8%
	N@5	0.2285	0.2114	0.2344	0.2202	0.2615	0.3459	0.3043	**0.3591**	3.8%
	N@10	0.2603	0.2472	0.2703	0.2635	0.2867	0.3867	0.3426	**0.3994**	3.3%
H&K	R@1	0.0660	0.0688	0.0860	0.0695	0.0809	0.1053	0.0973	**0.1158**	10.0%
	R@5	0.2058	0.2355	0.2467	0.1815	0.1914	0.2817	0.2585	**0.2984**	5.9%
	R@10	0.3041	0.3580	0.3610	0.2712	0.2717	0.3898	0.3696	**0.4099**	5.2%
	N@5	0.1368	0.1539	0.1592	0.1374	0.1403	0.1955	0.1796	**0.2096**	7.2%
	N@10	0.1686	0.1945	0.1976	0.1687	0.1677	0.2304	0.2149	**0.2454**	6.5%

4.1 Recommendation Performance

Table 3 shows the recommendation performance. R@1 and N@1 are the same and precision can be calculated based on recall for the user's next purchased item prediction [13]. The last column (Improv.) shows the percentage improvement of HierTrans over the next-best alternative. Overall, we observe the full-blown HierTrans improves upon all the baselines on all datasets in recall@k and NDCG@k. Concretely, HierTrans outperforms the next-best alternative by 7.02% in recall and 7.72% in NDCG on average.

Specifically, HierTrans consistently outperforms TransRec, which shows the importance of modeling item multi-relations inside user sequences and considering the T previous items. More importantly, comparing with TransFM, HierTrans consistently achieves a better performance. This confirms HierTrans can more effectively utilize item category-level information for sequential recommendation. For different datasets, HierTrans obtains a large improvement in Electronics and H&K while the improvement in C&A is relatively small. We attribute the good performance of HierTrans to the rich item relations in both Electronics and H&K based on the Relations column in Table 2. Furthermore, the proposed HierTrans outperforms all baselines on both sparse and dense datasets. One likely reason is that the incorporated item information in HierTrans can (1) provide more evidence for user sequential patterns and thus can alleviate the sparsity problem in user sequences; (2) also give more insights to effectively learn sequential patterns among the complex user interactions in dense datasets. We also check different latent dimensions ([50,100,150,200]). HierTrans consistently outperforms the other methods (omit this part due to space limitation).

4.2 Ablation Study

The section introduces several variants of HierTran to analyze their effects: (1) *TransRecC*: we only incorporate item category-level relations in TransRec without considering attention of user translations and multiple previous items; (2) *TransRecI*: we directly apply item-level relations in the user sequence graph rather than category-level; (3) *Concat*: instead of using $\vec{i}^* = \vec{i} + \vec{c}^i$, we concatenate the category embedding to item embedding; (4) *Connection*: we introduce "belongs to" relation embedding to connect $\vec{i}$ and $\vec{c}^i$; (5) *No Pre-train*: we forgo the pre-training phase for $\mathcal{G}^C$ and $\mathcal{G}^I$; (6) *No Attention*: we remove the attention (Equation (5)) and use the unified transition vectors; (7) *No Multi-Items*: Only the nearest recent item is used: $\vec{Head}(\mathcal{G}^I, \mathcal{G}^C|S_n^u) + \vec{Trans}(r_u|S_n^u, r_k) \approx \vec{S}_{n+1}^{*u}$; (8) *Item Attention/Convolution*: we use self-attention /CNN for $\vec{Head}(\cdot)$. Hyper-parameters are fine tuned based on the validation datasets.

The results are shown in Table 4. The *Default* row shows HierTrans results. We observe methods that consider item relations (such as TransRecC and TransRecI) outperform TransRec, which supports that (category-level) item relations play an important role in sequential recommendation. Specifically, TransRecI achieves higher performance than TransRecC. One likely reason is that there can be information leakage since the item-level relations in the Amazon dataset are captured from user purchase history [21]. We also argue that category-level item information is more general and stable for sequential recommendation.

Table 4: Ablation study on Electronics. Similar results hold for the other two datasets.

Setup	R@5	N@5	Setup	R@5	N@5
Default	0.4551	0.3285	No Pre-train	0.4249	0.3034
TransRecC	0.4150	0.2999	No Attention	0.4497	0.3232
TransRecI	0.4282	0.3122	No Multi-Items	0.4147	0.3016
Concat	0.3994	0.2845	Item Attention	0.4406	0.3161
Connection	0.3931	0.2825	Item Covolution	0.4517	0.3267

The way to incorporate item multi-relations and dynamic user interactions heavily impacts HierTrans performance. Comparing *Concat* and *Connection*, directly adding category embeddings achieves the best performance. Furthermore, results of *No Attention* and *No Multi-Items* show both the attention and multi-items consideration play an important role to improve sequential recommendation, which confirms our intuition that user sequence patterns are collaboratively influenced by both item relations and previous T purchased items. For the choice of $\vec{Head}(\cdot)$, we observe using average performs better than using attention/CNN here. One possible reason is that the three datasets are very sparse while attention/CNN brings many parameters, which makes the corresponding method easier to overfit.

5 CONCLUSION

We propose a novel hierarchical translation-based sequential recommendation that *adaptively* aggregates item multi-relations and dynamic user preferences from both a user's interacted item patterns and a user's dynamic translation behavior. Experiments on different datasets show HierTrans consistently outperforms state-of-the-art sequential recommenders. In the future, we are interested in exploring the influence of different types of item relations (e.g., movies with the same director) on *dynamic* user behaviors.

REFERENCES

[1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271* (2018).

[2] Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. 2014. A semantic matching energy function for learning with multi-relational data. *Machine Learning* 94, 2 (2014), 233–259.

[3] Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in Neural Information Processing Systems*. 2787–2795.

[4] Yixin Cao, Xiang Wang, Xiangnan He, Zikun Hu, and Tat-Seng Chua. 2019. Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In *International Conference on World Wide Web*. 151–161.

[5] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* (2014).

[6] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2017. Translation-based recommendation. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*. 161–169.

[7] Ruining He, Wang-Cheng Kang, and Julian McAuley. 2018. Translation-based Recommendation: A Scalable Method for Modeling Sequential Behavior. In *IJCAI*. 5264–5268.

[8] Ruining He and Julian McAuley. 2016. Fusing similarity models with markov chains for sparse sequential recommendation. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*. IEEE, 191–200.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*. 173–182.

[10] Yun He, Yin Zhang, Weiwen Liu, and James Caverlee. 2020. Consistency-Aware Recommendation for User-Generated Item List Continuation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 250–258.

[11] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. 2015. Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939* (2015).

[12] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[13] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *2018 IEEE International Conference on Data Mining (ICDM)*. IEEE, 197–206.

[14] Wang-Cheng Kang, Mengting Wan, and Julian McAuley. 2018. Recommendation through mixtures of heterogeneous item relationships. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1143–1152.

[15] Yehuda Koren. 2008. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. 426–434.

[16] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. DBpedia–a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web* 6, 2 (2015), 167–195.

[17] Yankai Lin, Zhiyuan Liu, Huanbo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015. Modeling relation paths for representation learning of knowledge bases. *arXiv preprint arXiv:1506.00379* (2015).

[18] Yankai Lin, Zhiyuan Liu, Maosong Sun, Yang Liu, and Xuan Zhu. 2015. Learning entity and relation embeddings for knowledge graph completion. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[19] Weizhi Ma, Min Zhang, Yue Cao, Woojeong Jin, Chenyang Wang, Yiqun Liu, Shaoping Ma, and Xiang Ren. 2019. Jointly learning explainable rules for recommendation with knowledge graph. In *International Conference on World Wide Web*. 1210–1221.

[20] Julian McAuley, Rahul Pandey, and Jure Leskovec. 2015. Inferring networks of substitutable and complementary products. In *Proceedings of the 21th ACM SIGKDD international conference on Knowledge discovery and data mining*. 785–794.

[21] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 43–52.

[22] Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2012. Factorizing yago: scalable machine learning for linked data. In *Proceedings of the 21st International Conference on World Wide Web*. 271–280.

[23] Rajiv Pasricha and Julian McAuley. 2018. Translation-based factorization machines for sequential recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 63–71.

[24] Vineeth Rakesh, Suhang Wang, Kai Shu, and Huan Liu. 2019. Linked variational autoencoders for inferring substitutable and supplementary items. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 438–446.

[25] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *UAI*. AUAI Press.

[26] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. 2010. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings of the 19th International Conference on World Wide Web*. 811–820.

[27] 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*. 565–573.

[28] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 417–426.

[29] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*. 165–174.

[30] Zihan Wang, Ziheng Jiang, Zhaochun Ren, Jiliang Tang, and Dawei Yin. 2018. A path-constrained framework for discriminating substitutable and complementary products in e-commerce. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. 619–627.

[31] Zhigang Wang and Juan-Zi Li. 2016. Text-Enhanced Representation Learning for Knowledge Graph. In *IJCAI*. 1293–1299.

[32] Zhen Wang, Jianwen Zhang, Jianlin Feng, and Zheng Chen. 2014. Knowledge graph embedding by translating on hyperplanes. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

[33] Xin Xin, Xiangnan He, Yongfeng Zhang, Yongdong Zhang, and Joemon Jose. 2019. Relational collaborative filtering: Modeling multiple item relations for recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 125–134.

[34] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*. 1480–1489.

[35] Rex Ying, Ruining He, Kaifeng Chen, Pong Eksombatchai, William L Hamilton, and Jure Leskovec. 2018. Graph convolutional neural networks for web-scale recommender systems. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge discovery and data mining*. 974–983.

[36] Jiaxuan You, Yichen Wang, Aditya Pal, Pong Eksombatchai, Chuck Rosenburg, and Jure Leskovec. 2019. Hierarchical temporal convolutional networks for dynamic recommender systems. In *International Conference on World Wide Web*. 2236–2246.

[37] Fajie Yuan, Alexandros Karatzoglou, Ioannis Arapakis, Joemon M Jose, and Xiangnan He. 2019. A simple convolutional generative network for next item recommendation. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*. 582–590.

[38] Yongfeng Zhang, Qingyao Ai, Xu Chen, and Pengfei Wang. 2018. Learning over knowledge-base embeddings for recommendation. *arXiv preprint arXiv:1803.06540* (2018).

[39] Yin Zhang, Haokai Lu, Wei Niu, and James Caverlee. 2018. Quality-aware neural complementary item recommendation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 77–85.