Explainable meta-path based recommender systems

Article

Accepted Version

Explainable Meta-Path Based Recommender Systems

THANET MARKCHOM, University of Reading, United Kingdom
HUIZHI LIANG, Newcastle University, United Kingdom
JAMES FERRYMAN, University of Reading, United Kingdom

Meta-paths have been popularly used to provide explainability in recommendations. Although long/complicated meta-paths could represent complex user-item connectivity, they are not easy to interpret. This work tackles this problem by introducing a meta-path translation task. The objective is to translate a meta-path to its comparable explainable meta-paths that perform similarly in terms of recommendation but have higher explainability compared to the given one. We propose a definition of meta-path explainability to determine comparable explainable meta-paths and a meta-path grammar that allows comparable explainable meta-paths to be formed in a similar way as sentences in human languages. Based on this grammar, we propose a meta-path translation model, a sequence-to-sequence (Seq2Seq) model to translate a long and complicated meta-path to its comparable explainable meta-paths. Two novel datasets for meta-path translation were generated based on two real-world recommendation datasets. The experiments were conducted on these generated datasets. The results show that our model outperformed state-of-the-art Seq2Seq baselines regarding meta-path translation and maintained a better trade-off between accuracy and diversity/readability in predicting comparable explainable meta-paths. These results indicate that our model can effectively generate a group of explainable meta-paths as alternative explanations for those recommendations based on any given long/complicated meta-path.

CCS Concepts: • **Information systems → Recommender systems**; *Language models*; • **Computing methodologies →** *Machine translation.*

Additional Key Words and Phrases: Heterogeneous Information Network, Meta-Path, Explainability, Datasets

ACM Reference Format:
Thanet Markchom, Huizhi Liang, and James Ferryman. 2023. Explainable Meta-Path Based Recommender Systems. 1, 1 (October 2023), 29 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The amount of information available for being chosen by users has been exponentially increasing during the past decade. To avoid bombarding users with a high volume of information, recommender systems have been developed. These systems select pieces of information or items of interest for users based on user profiles, item metadata, and their historical interactions. Many recommender systems are "black-box models" in which their decision-making mechanisms are not transparent. This results in issues of understanding how they work or justifying the recommendations obtained from these systems. Some recent regulations such as the General Data Protection Regulation (GDPR) of the European Union and other countries [13] require artificial intelligence systems to

Authors' addresses: Thanet Markchom, Department of Computer Science, University of Reading, United Kingdom, t.markchom@pgr.reading.ac.uk; Huizhi Liang, School of Computing, Newcastle University, United Kingdom, huizhi.liang@ncl.ac.uk; James Ferryman, Department of Computer Science, University of Reading, United Kingdom, j.m.ferryman@reading.ac.uk.

be transparent or explainable to ensure fairness and trust for the users [8, 10]. According to this, explainability has become another focal point for developing modern recommender systems.

Explainable recommender systems are capable of providing explanations of why the items are recommended. Such explanations can help users make decisions to accept or reject the recommendation, knowing that it was based on their preferences or needs [52]. For example, if a user receives a recommendation but the explanation reveals that it was made based on its popularity rather than the user's interests, the user may select other recommendations that are more relevant to their interests. Apart from this, they can also increase the persuasiveness of recommendations and help maintain the systems [36]. Several resources have been utilized to achieve both accuracy and explainability in recommendations including heterogeneous information networks. A heterogeneous information network (HIN) is a directed graph whose nodes are entities (e.g., users, items, categories, or brands) and whose edges are relations between entities. Considering HINs, there are two types of relations that can be observed: single-hop relations and multi-hop relations. Single-hop relations are explicit and direct relations between two adjacent nodes in a network. Such relations are certainly useful but they can only provide shallow insights. On the other hand, multi-hop relations are relations between two non-adjacent nodes that are implicitly connected to each other through a sequence of nodes and relations (path). In general, multi-hop relations are highly informative. They are capable of providing new and unforeseen insights within a HIN that single-hop relations cannot provide. HIN-based recommender systems leverage these multi-hop relations to model high-order connectivity between nodes in HINs. Many models have been used to extract this information, for instance, Gated Graph Neural Network (GGNN) [43], Graph Convolutional Neural Network (GCNN) [47] and Graph Attention Network [40]. These models are highly effective for extracting multi-hop relations but they typically lack explainability since it is difficult to examine the semantic meanings of the extracted multi-hop relations [14].

Meta-paths have been proposed to extract semantically meaningful multi-hop relations in a HIN under certain assumptions [14]. A meta-path based recommender system can generate recommendations based on specific meta-paths and use these meta-paths as explanations [23, 26, 41]. Depending on the circumstances, different meta-paths with variations in lengths and complexities can be utilized to achieve intended performances. Some prior studies have shown that long and complicated meta-paths can be useful and more informative than shorter meta-paths [5, 24, 46, 50]. However, the explainability of long and complicated meta-paths is intuitively low. Explaining recommendations with these meta-paths could be more difficult to understand than shorter and less complex ones. For example, given a HIN in Figure 1, let U, P, C and B denote user, item, category, and brand node types respectively. In this figure, a meta-path based recommender system based on meta-path $UPBPCP$ recommends "Item A" to "User 2" based on $UPBPCP$. It can be interpreted as "Item A is recommended to User 2 because it is in the same category as an item (Item B) that has the same brand as User 2's item (Item C)". Similarly, a meta-path based recommender system based on $UPCP$ also recommends "Item A" to "User 2". The explanation is based on $UPCP$, i.e., "Item A is recommended to User 2 because it is in the same category as User 2's item (Item C)". Although both meta-paths can be used to explain this recommendation, the explanation based on $UPCP$ is more interpretable than $UPBPCP$, as it is shorter. Hence, generating more comprehensible explanations for those recommendations could lead to better explainability in meta-path based recommender systems.

To bridge the gap of better explaining meta-path based recommendations, we propose a method to find more comprehensible explanations for meta-path based recommendations. First, we propose how to quantify **meta-path explainability** based on three aspects, i.e., readability, credibility, and diversity. Then, given any meta-path, we define **comparable explainable meta-paths** as meta-paths that perform recommendation similarly but have higher explainability compared to the given

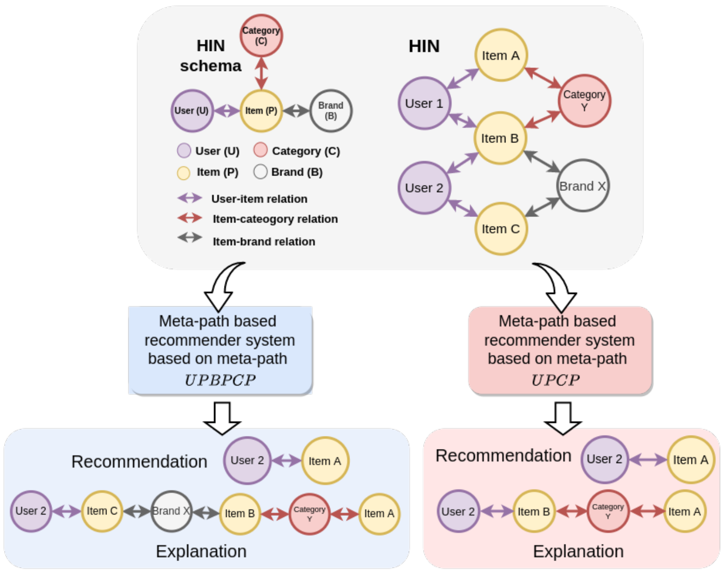


Fig. 1. Toy example of explaining recommendations obtained from meta-path based recommender systems using meta-path $UPBPCP$ and meta-path $UPCP$

meta-path. They can be used as more comprehensible explanations for those recommendations based on the given meta-path. Depending on the number of candidate meta-paths, identifying comparable explainable meta-paths manually can be time-consuming. Therefore, we propose a method for generating comparable explainable meta-paths. Considering meta-paths as languages, we assume that they can be constructed based on certain grammar rules. We define a **meta-path grammar** based on the concept of quasi-synchronous context-free grammar [32]. With this grammar, we propose a **meta-path translation model**, a probabilistic model that maps a meta-path to its comparable explainable ones. Since our task is similar to a machine translation in natural language processing (NLP), a sequence-to-sequence (Seq2Seq) approach is adopted to build a meta-path translation model. The proposed meta-path translation model consists of three parts. The first part is the *parser* for generating the parse tree of a source meta-path. The second part is the *encoder* which extracts latent features of node types in a source meta-path and hierarchical-structure information from a source meta-path simultaneously. The last part is the *decoder*, which is a modified Seq2Seq model based on latent neural grammar [19]. It maps a source meta-path to a target meta-path based on the meta-path grammar. Unlike most existing Seq2Seq models, the proposed model is capable of modeling non-hierarchical and hierarchical dependencies between node types in a meta-path. Also, it is suitable for our problem which is a one-to-many task compared to other Seq2Seq models. To the best of our knowledge, this is the first work that considers meta-paths from a linguistic point of view. The contributions of this work are summarized as follows:

- We introduce the meta-path translation task for recommender systems. The goal is to find the corresponding comparable explainable meta-paths of a long and complicated one. We propose how to quantify the explainability of a meta-path. Based on this explainability, we can then identify comparable explainable meta-paths of any given meta-path.
- Two meta-path translation datasets for meta-path based recommendations were generated based on two real-world datasets. In these generated datasets, each input is a meta-path

and each corresponding output is a group of its comparable explainable meta-paths. These datasets are publicly available. They are the first of this type of dataset.
- To translate a meta-path, we present a novel view of meta-paths and how they are formed from a grammatical perspective. We define a meta-path grammar (a set of rules) that allows us to systematically construct explainable meta-paths in a similar way as sentences in human languages.
- Based on the meta-path grammar, we propose a meta-path translation model. The key idea is to use the latent neural grammar based on the meta-path grammar defined in our work. Also, both non-hierarchical and hierarchical structures of a meta-path are considered simultaneously to effectively learn the translation.
- We conducted extensive experiments on real-world datasets for generating more comprehensible explanations for the meta-path based recommendations.

2 RELATED WORK

Collaborative filtering (CF) is one of the popular recommender system approaches using implicit/explicit user-item interactions to make recommendations based on the similarity between users [51]. Several CF-based models have been proposed, e.g., CF-KNN that uses the K-Nearest-Neighbor method (KNN) for measuring similarity between users [2], Matrix Factorization (MF) model [20] that decomposes a user-item interaction matrix into low-dimensional user/item latent factors and BPR-MF model [28] that combines MF with Bayesian Personalized Ranking (BPR) scheme. Since these models rely on only user-item interactions, they typically suffer from the lack of user-item interactions (cold start problem) [30].

HINs have been widely used to resolve the cold start problem in recommendation [14, 18, 38, 40]. They provide auxiliary information in the form of multi-hop relations between entities in a HIN. Such high-order relations can be accessed and utilized by many methods, e.g., Graph Convolutional Neural Network (GCNN) [47], RippleNet [38], and Graph Attention Network [40]. Recently, meta-paths have become a powerful tool for leveraging multi-hop relations in HINs [14]. They can extract various multi-hop relation types between user-item pairs with different semantic assumptions. Most meta-path based recommender systems use meta-path based contexts to model the semantic connectivity between users and items. For example, PathSim [33] method was applied on meta-path based contexts to compute the semantic connectivity in [49]. Such connectivity was then used to enrich the user-item interaction matrix. In [24], metapath2vec [9] method was also applied to meta-path based contexts to generate user and item representations for a CF-KNN model. Both PathSim and metapath2vec methods were used in [26] to model the user-item semantic similarities for a BPR model. In [6], the various meta-path based contexts were attentively combined to learn user and item representations. Some models learned meta-path representations and integrated them into the learning frameworks. For instance, a co-attention mechanism was used to learn user, item, and meta-path representations in [18]. The attention weights were used to identify the most important meta-path. In [17], user, item, and relation representations were learned to capture the probabilities of meta-path based interactions between user-item pairs. Depending on the meta-path used in these methods, the generated user and item representations reflect how similar they are under the meaningful assumptions [9, 14, 33]. Such assumptions can be used to explain the recommendations based on these user and item representations. However, if the meta-path is long and complicated, the users may not fully understand the meaning behind it. Therefore, translating such meta-paths to more explainable ones could provide more comprehensible explanations for those recommendations.

The task of meta-path translation can be formulated as a sequence-to-sequence (Seq2Seq) task where a sequence of nodes and relations is mapped into another sequence of nodes and relations that is more interpretable. As a result, this task can be achieved by using a Seq2Seq model. Typically, a Seq2Seq model is applied to accomplish a Natural Language Processing (NLP) task, e.g., machine translation [39], text summarization [31], and style transfer [11]. Some architectures such as Recurrent Neural Networks (RNNs) [34], Convolutional Neural Networks (CNNs) [12] and Transformer [37] have been adopted to build Seq2Seq models. They are effective for modeling latent features between components in a sequence regardless of its hierarchical structure [25]. Recently, a Seq2Seq model using latent neural grammars was proposed [19]. This model applies the concept of probabilistic quasi-synchronous context-free grammars (QCFGs) on the source and target sequences allowing a Seq2Seq translation to be done in a hierarchical and grammatical way. Since meta-paths are sequences, a Seq2Seq model is intuitively suitable for the task of meta-path translation. However, such context-free grammars have only been used in the NLP area. There is still a gap in using context-free grammars in the recommendation area, especially for the explainable recommendation task. To fill the gap, adopting an existing Seq2Seq model to solve the task is possible. However, existing models either rely on a non-hierarchical structure or a hierarchical structure of a sequence. Simultaneously leveraging both of them could lead to better performance for this task.

3 EXPLAINABLE META-PATH

In this section, we discuss the definitions of HIN, meta-path, and explainable meta-path proposed in this work. Then, we explain how to obtain relatively more explainable meta-paths given a long and complicated one.

Definition 1. ***(HIN schema)*** *Let $\mathbb{G}$ denote a HIN schema, $\mathbb{G} = (\mathbb{N}, \mathbb{R}, \mathbb{W})$ where $\mathbb{N}$ is a set of node types, $\mathbb{R}$ is a set of relation types and $\mathbb{W} : \mathbb{R} \longrightarrow \mathfrak{R}$ is a non-negative weight function mapping each relation to a non-negative weight in $\mathfrak{R}$. Let $N_i, N_j \in \mathbb{N}$ be any two node types, $R_{N_i,N_j} \in \mathbb{R}$ denotes the relation type connecting from a node with node type N_i to a node with node type N_j. For any $R_{N_i,N_j} \in \mathbb{R}$, let $R^{-1}_{N_i,N_j}$ denote an inverse relation type from N_j to N_i.*

Definition 2. ***(HIN)*** *Given a HIN schema $\mathbb{G} = (\mathbb{N}, \mathbb{R}, \mathbb{W})$, a HIN is defined as a directed graph $\mathcal{G} = (\mathcal{N}, \mathcal{R})$ where $\mathcal{N}$ is a set of nodes and $\mathcal{R}$ is a set of relations. Each node and relation is associated with their type mapping function: $\phi : \mathcal{N} \longrightarrow \mathbb{N}$ and $\psi : \mathcal{R} \longrightarrow \mathbb{R}$ respectively. Given nodes $x, y \in \mathcal{N}$, $r_{x,y} \in \mathcal{R}$ denotes a relation from x to y and its weight is denoted by $w(r_{x,y}) = \mathbb{W}(\psi(r_{x,y}))$.*

An example of a HIN schema is shown in Figure 1. In this schema, there are 4 node types, i.e., user (U), item (P), category (C), and brand (B) node types, and 6 relation types, i.e., user-item, item-category and item-brand relation types including their inverse relation types. An example of a HIN with this schema is also provided in this figure.

Definition 3. ***(Meta-Path)*** *Given a HIN $\mathcal{G}$ with a schema $\mathbb{G}$, a meta-path m is defined as $N_1 \xrightarrow{R_{N_1,N_2}} N_2 \cdots N_l \xrightarrow{R_{N_l,N_{l+1}}} N_{l+1}$ (abbreviated as $N_1N_2 \cdots N_{l+1}$), describes a composite relation $R_{N_1,N_2} \circ \cdots \circ R_{N_l,N_{l+1}}$ between node type N_1 and N_{l+1} where $\circ$ is the composition operator on relations. l is the length of m, $\mathbb{N}_m = \{N_1, N_2, ..., N_{l+1}\}$ is a set of node types in m and $\mathbb{R}_m = \{R_{N_1,N_2}, \cdots, R_{N_l,N_{l+1}}\}$ is a set of relation types in m.*

A meta-path provides meaningful high-order connectivity between users and items for learning recommendations. Due to its semantic meaning, it can be used as an explanation of why the items are recommended. For example, let U, P, and C denote the user node type, item node type, and category node type respectively. Any recommendations based on $UPCP$ can be explained that they are recommended since they have the same categories as the users' previously interacted items. Intuitively, some meta-paths are perplexing and hard to understand, especially those that are long and contain various node types. Nonetheless, these meta-paths are still useful for capturing diversity in users' preferences [24].

We customize three metrics in [45] that are commonly used to measure sequence (or path) readability, credibility, and diversity to quantify the explainability of a meta-path. Let m be a meta-path. *Meta-path readability* $\mathrm{R}(m)$ is computed by

$$\mathrm{R}(m) = \frac{1}{\sqrt{l \cdot |\mathbb{N}_m|}} \tag{1}$$

It is inversely proportional to the length of m and the number of node types in m. The readability decreases as the length and the total number of node types increase. *Meta-path credibility* $\mathrm{C}(m)$ is computed by

$$\mathrm{C}(m) = \Pi_{i=1}^{l} \mathbb{W}(R_{N_i,N_{i+1}}) \tag{2}$$

where $\mathbb{W}(R_{N_i,N_{i+1}})$ is the weight of relation $R_{N_i,N_{i+1}}$ in m. It is the accumulated weight of all relation weights in the meta-path. The higher credibility means that the meta-path is more accountable based on pre-defined weights in the schema. *Meta-path diversity* $\mathrm{D}(m)$ is defined as

$$\mathrm{D}(m) = \log_{l+1} |\mathbb{R}_m| \tag{3}$$

Some studies have shown that relation sequences with diversity are more comprehensive and persuasive for humans [1, 45]. Thus, the higher the diversity of m is, the more explainable.

Definition 4. ***Explainable Meta-Path*** *Given a meta-path m, let δ_R, δ_C, and δ_D denote the thresholds for readability, credibility, and diversity respectively. If* $\mathrm{R}(m) > \delta_R$, $\mathrm{C}(m) > \delta_C$ *and* $\mathrm{D}(m) > \delta_D$, *then m is explainable.*

We define the criteria to determine which meta-paths are comparable but more explainable given a long and complicated meta-path. The key idea is to find a group of explainable meta-paths that consist of the same node types and perform as well as a given meta-path. These conditions are summarized in the following definition:

Definition 5. ***Comparable Explainable Meta-Path*** *Given a meta-path m, a comparable explainable meta-path is a meta-path m' satisfying these conditions:*

- *m' is an explainable meta-path*
- $\mathbb{N}_{m'} \subseteq \mathbb{N}_m$
- $l' \leq k < l$
- $|\mathrm{A}(m) - \mathrm{A}(m')| < \delta$

where $\mathbb{N}_{m'}$ is the set of node types of m', l' is the length of m', k is the maximum length for selecting short/interpretable meta-paths, $\mathrm{A}(m)$ *and* $\mathrm{A}(m')$ *are any performance evaluation values (e.g., Hit ratio, Precision, or Recall) based on m and m' respectively, and δ is the pre-defined performance evaluation threshold. For any m, there can be multiple meta-paths corresponding to these conditions.*

Based on this definition, we can find more explainable meta-paths (m') to explain recommendations based on a long and complicated meta-path (m). However, the number of possible explainable meta-paths is directly proportional to the length of m and the number of unique node types in m. The higher the length and the number of node types are, the larger the number of explainable meta-paths that could be comparable. Thus, instead of considering the entire search space, we propose a method to find comparable explainable meta-paths for recommendations based on long/complicated meta-paths.

4 META-PATH TRANSLATION MODEL

In this section, we discuss the proposed model to translate or map a given meta-path to its comparable explainable meta-paths.

4.1 Problem Formulation

Let $\mathcal{M}$ be a set of meta-paths and $\mathcal{M}'$ be a set of explainable meta-paths. Given a meta-path $m \in \mathcal{M}$, the problem is to find a function $f : \mathcal{M} \rightarrow \mathrm{P}(\mathcal{M}')$ that maps m to a subset of comparable explainable meta-paths of m where $\mathrm{P}(\mathcal{M}')$ denotes the power set of $\mathcal{M}'$. This problem can be considered as a one-to-many task in which an input meta-path yields a set of its comparable explainable meta-paths.

4.2 Meta-Path Grammar

To find f, we first assume properties of meta-paths in $\mathcal{M}$ and $\mathcal{M}'$. Inspired by the concept of context-free grammar in NLP, we assume that meta-paths in $\mathcal{M}$ and $\mathcal{M}'$ follow certain grammars. These grammars represent certain rules of how meta-paths in both $\mathcal{M}$ and $\mathcal{M}'$ are constructed. Based on Definition 5, each $m' \in \mathcal{M}'$ is related to its corresponding $m \in \mathcal{M}$. Therefore, the grammar of m' should also depend on its corresponding m. To capture this property, we adopt the concept of *quasi-synchronous context-free grammar* [32] to define the grammar of meta-paths in $\mathcal{M}$ and $\mathcal{M}'$.

Definition 6. ***Meta-Path Quasi-Synchronous Context-Free Grammar*** *Given a meta-path m as a source meta-path and m' as a target meta-path. Let t and t' denote the parse tree of m and m' respectively. A meta-path quasi-synchronous context-free grammar (QCFG) is represented as*

$$\mathbb{G}[t] = (\mathbb{S}, \mathbb{N}, \mathbb{P}, \mathbb{E}, \mathbb{R}[t], \theta) \tag{4}$$

where $\mathbb{S}$ is the distinguished start symbol, $\mathbb{N}$ is the set of non-terminals that expand to other non-terminals, $\mathbb{P}$ is the set of non-terminals that expand to terminals (i.e. pre-terminals), $\mathbb{E}$ is the set of terminals which is the set of node types, and $\mathbb{R}[t]$ is a set of context-free rules conditioned on the source tree t, where each rule follows one of these following rules:

$$\mathbb{S} \rightarrow \mathsf{A}[\alpha_i], \mathsf{A} \in \mathbb{N}, \alpha_i \subseteq t \tag{5}$$

$$\mathsf{A}[\alpha_i] \rightarrow \mathsf{B}[\alpha_j]\mathsf{C}[\alpha_k], \mathsf{A} \in \mathbb{N}, \mathsf{B}, \mathsf{C} \in \mathbb{N} \cup \mathbb{P}, \alpha_i, \alpha_j, \alpha_k \subseteq t \tag{6}$$

$$\mathsf{D}[\alpha_i] \rightarrow w, \mathsf{D} \in \mathbb{P}, w \in \mathbb{E}, \alpha_i \subseteq t \tag{7}$$

where α_i's are subsets of nodes in the source tree t, and θ is the parameters of the rule probabilities $p_\theta(r)$ for each $r \in \mathbb{R}[t]$. These subsets of nodes are aligned with certain nodes in the target tree t' when performing the translation. Note that B *and* C *are arbitrary non-terminals or pre-terminals in* $\mathbb{N} \cup \mathbb{P}$. D *is an arbitrary pre-terminal in* $\mathbb{P}$. *If* B *(or* C*) is a pre-terminal in* $\mathbb{P}$, *it is equivalent to* D *in Eq. 7 and can expands to a terminal $w \in \mathbb{E}$ following the grammar rule* $\mathsf{B}[\alpha_i] \rightarrow w$ *(or* $\mathsf{C}[\alpha_i] \rightarrow w$*).*

Figure 2 shows an example of meta-path grammar and how the target tree nodes are aligned with certain nodes in the source tree in order to translate the source meta-path *UPBPCP* to the target meta-path *UPCP*. Let t be the parse tree of the source meta-path *UPBPCP* (i.e., the source tree) and t' be the parse tree of the target meta-path *UPCP* (i.e., the target tree). $\alpha_0, \alpha_1, ..., \alpha_{10}$ denote non-terminal nodes of the source tree while A, B, C, and D denote non-terminals of the target tree. The node types U, P, B, P, C and P in blue are terminals nodes of t while the node types U, P, C, and P in red are terminal nodes of t'. Each non-terminal node in the target tree t' is transduced by certain nodes in the source tree t and is parsed into other non-terminals or terminals. For instance, A is transduced by α_{10} in the source tree. Then, A is parsed into B which is transduced by α_9 and D which is transduced by α_5. Since both B and D are non-terminals, they, therefore, have to be parsed

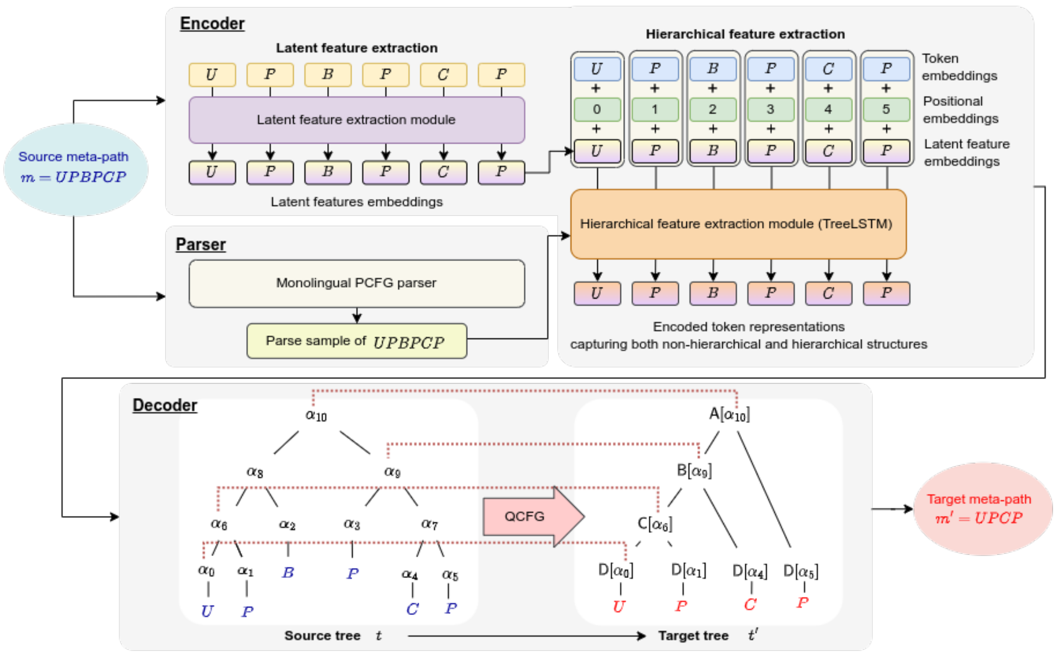

Encoder
Latent feature extraction
Latent feature extraction module
Latent features embeddings
Hierarchical feature extraction
Token embeddings
Positional embeddings
Latent feature embeddings
Hierarchical feature extraction module (TreeLSTM)
Encoded token representations capturing both non-hierarchical and hierarchical structures
Source meta-path $m = UPBPCP$
Parser
Monolingual PCFG parser
Parse sample of $UPBPCP$
Decoder
QCFG
Source tree t
Target tree t'
Target meta-path $m' = UPCP$

Fig. 2. The proposed meta-path translation model

again. B is parsed into two non-terminals while D is parsed into a terminal node P. By running this process from the root node until reaching all non-terminal nodes, the target tree t' which produces the target meta-path $UPCP$ is formed as an output for this meta-path translation.

4.3 Meta-Path Translation Model

To build a meta-path translation model, we adopt a Seq2Seq model with latent neural grammar from [19]. This model was originally proposed for NLP tasks such as machine translation and style transfer. It is capable of capturing the hierarchical structure of a sequence based on any latent QCFG. However, latent features extracted from the non-hierarchical structure of a sequence can be beneficial as well. A non-hierarchical structure is a structure of a sequence that is not in a level-like or tree-like form based on a specific grammar of the sequence. It represents local and global dependencies between components in a sequence regardless of the sequence grammar. Relying only on a hierarchical structure, such dependencies are neglected. Considering both non-hierarchical and hierarchical structures of a sequence can lead to a more effective meta-path translation model. We, therefore, adopt this method and modify it to consider non-hierarchical and hierarchical structures simultaneously.

The proposed meta-path translation model consists of three parts: (1) **parser** that finds the parse tree of a source meta-path, (2) **encoder** that encodes the dependency of tokens (node types) in a source meta-path in both non-hierarchical and hierarchical perspectives, and (3) **decoder** that uses the source parse tree and the source token embeddings from the encoder to translate a source meta-path based on the meta-path grammar. Figure 2 illustrates the proposed meta-path translation model given a source meta-path $UPBPCP$ and its target meta-path $UPCP$.

Parser. Typically, there exists a well-defined parser that can be applied to sentences in natural languages to extract parse trees of these sentences immediately. However, in our case, there is no parser developed specifically for meta-paths. Therefore, we adopt the same approach in [19] to jointly train the parser with the encoder and the decoder. Following [19], a *monolingual PCFG* with parameters ϕ is adopted to find the distribution of the source tree t given the source meta-path m denoted as $p_\phi(t|m)$. Therefore, given a meta-path m, the source tree t of m can be sampled from $p_\phi(t|m)$. This source tree is then used as an input for the encoder in the next step.

Encoder. The encoder consists of *latent feature extraction module* and *hierarchical feature extraction module*. The latent feature extraction module is firstly used to obtain dependency information of node types (tokens) in a source meta-path regardless of its hierarchical structure. This module can be any type of architecture such as LSTM, CNN, or Transformer. It takes a source meta-path as an input and outputs the *latent feature embeddings* of tokens in a source meta-path. In this way, the non-hierarchical-structure information is captured within these extracted latent feature embeddings. These latent feature embeddings are used as inputs for the hierarchical feature extraction module. They can be considered prior knowledge for hierarchical feature extraction.

As for the hierarchical feature extraction, following [19], TreeLSTM [35] is used to encode the hierarchical structure of a source meta-path given its source tree from the parser. In their work, only the token embeddings were used as inputs of TreeLSTM. However, in our case, the same node types repeatedly appear in a meta-path, for instance, meta-path *UPBPCP* has three *P* node types in three different positions. This could lead to confusion in learning the dependency information. Thus, we add the *positional embeddings* along with the token embeddings to differentiate such repeating tokens based on their positions. A positional embedding is added to each token to indicate its position in the meta-path. Altogether, the input embeddings of TreeLSTM are the sum of the token embeddings, the positional embeddings, and the latent feature embeddings. In this way, TreeLSTM generates final source token embeddings that encode both non-hierarchical-structure and hierarchical-structure information of a source meta-path.

Given a parse tree, for any node α_i in this tree, let N be the total number of child nodes of α_i and $\mathbf{h}_{\alpha_i k}$ and $\mathbf{c}_{\alpha_i k}$ be the hidden state vector and memory cell vector of its kth child in the tree respectively. The Tree-LSTM transition equations with the sum of the token embeddings, the positional embeddings, and the latent feature embeddings as input are as follows:

$$\mathbf{z}_{\alpha_i} = \boldsymbol{\sigma}\left(\mathbf{W}^{(\mathrm{z})}(\mathbf{x}_{\alpha_i} + \mathbf{x}^*_{\alpha_i} + \mathbf{x}'_{\alpha_i}) + \sum_{n=1}^{N} \mathbf{U}^{(\mathrm{z})}_{n}\mathbf{h}_{\alpha_i n} + b^{(\mathrm{z})}\right) \tag{8}$$

$$\mathbf{f}_{\alpha_i k} = \boldsymbol{\sigma}\left(\mathbf{W}^{(\mathrm{f})}(\mathbf{x}_{\alpha_i} + \mathbf{x}^*_{\alpha_i} + \mathbf{x}'_{\alpha_i}) + \sum_{n=1}^{N} \mathbf{U}^{(\mathrm{f})}_{kn}\mathbf{h}_{\alpha_i n} + b^{(\mathrm{f})}\right) \tag{9}$$

$$\mathbf{o}_{\alpha_i} = \boldsymbol{\sigma}\left(\mathbf{W}^{(\mathrm{o})}(\mathbf{x}_{\alpha_i} + \mathbf{x}^*_{\alpha_i} + \mathbf{x}'_{\alpha_i}) + \sum_{n=1}^{N} \mathbf{U}^{(\mathrm{o})}_{n}\mathbf{h}_{\alpha_i n} + b^{(\mathrm{o})}\right) \tag{10}$$

$$\mathbf{v}_{\alpha_i} = \tanh\left(\mathbf{W}^{(\mathrm{v})}(\mathbf{x}_{\alpha_i} + \mathbf{x}^*_{\alpha_i} + \mathbf{x}'_{\alpha_i}) + \sum_{n=1}^{N} \mathbf{U}^{(\mathrm{v})}_{n}\mathbf{h}_{\alpha_i n} + b^{(\mathrm{v})}\right) \tag{11}$$

$$\mathbf{c}_{\alpha_i} = \mathbf{z}_{\alpha_i} \odot \mathbf{v}_{\alpha_i} + \sum_{k=1}^{N} \mathbf{f}_{\alpha_i k} \odot \mathbf{c}_{\alpha_i k} \tag{12}$$

$$\mathbf{h}_{\alpha_i} = \mathbf{o}_{\alpha_i} \odot \tanh(\mathbf{c}_{\alpha_i}) \tag{13}$$

where $\mathbf{x}_{\alpha_i}$, $\mathbf{x}^*_{\alpha_i}$ and $\mathbf{x}'_{\alpha_i}$ denote the token embedding, the positional embedding and the latent feature embedding of α_i respectively, $\boldsymbol{\sigma}$ denotes the logistic sigmoid function and $\mathbf{h}_{\alpha_i}$ is the hidden state vector of α_i which is used as the embedding of α_i for the decoder in the next step.

Decoder. After obtaining the source tree and the source token embeddings, the next step is to transduce the source tree t to the target tree t' via the decoder based on a QCFG. For each rule $r \in \mathbb{R}[t]$, the rule probability $p_\theta(r)$ is computed by either one of these following formulas:

$$p_\theta(\mathbb{S} \rightarrow \mathsf{A}[\alpha_i]) = \sigma(\mathbf{u}_{\mathbb{S}}^T \mathbf{e}_{\mathsf{A}[\alpha_i]}) \tag{14}$$

$$p_\theta(\mathsf{A}[\alpha_i] \rightarrow \mathsf{B}[\alpha_j]\mathsf{C}[\alpha_k]) = \sigma(f_1(\mathbf{e}_{\mathsf{A}[\alpha_i]})^T f_2(\mathbf{e}_{\mathsf{B}[\alpha_j]}) + f_3(\mathbf{e}_{\mathsf{C}[\alpha_k]})) \tag{15}$$

$$p_\theta(\mathsf{D}[\alpha_i] \rightarrow w) = \sigma(f_4(\mathbf{e}_{\mathsf{D}[\alpha_i]})^T \mathbf{u}_w + b_w) \tag{16}$$

where σ denote the softmax function, $\mathbf{u}_{\mathbb{S}}$ is an embedding of $\mathbb{S}$ randomly initialized, f_1, f_2, f_3 and f_4 are feedforward neural networks with residual layers, $\mathbf{e}_{\mathsf{A}[\alpha_i]}$ is an embedding of $\mathsf{A}[\alpha_i]$ computed by

$$\mathbf{e}_{\mathsf{A}[\alpha_i]} = \mathbf{u}_{\mathsf{A}} + \mathbf{h}_{\alpha_i} \tag{17}$$

where $\mathbf{u}_{\mathsf{A}}$ is an embedding of A randomly initialized and $\mathbf{h}_{\alpha_i}$ is the token embedding of α_i obtained from the encoder, $\mathbf{u}_w$ is a terminal node embedding and b_w is a terminal bias. These probabilities altogether define the probability of the target tree t' given the source tree t denoted as $p_\theta(t'|t)$.

With both $p_\phi(t|m)$ and $p_\theta(t'|t)$. The log marginal likelihood of a target meta-path m' given a source meta-path m with a QCFG $\mathbb{G}[t]$ is

$$\log p(m'|m) = \log \Big(\sum_{t \in \mathcal{T}(m)} \sum_{t' \in \mathcal{T}(m')} p_\theta(t'|t) p_\phi(t|m) \Big) \tag{18}$$

where $\mathcal{T}(m)$ and $\mathcal{T}(m')$ denote the sets of trees whose yields are m and m' respectively. Since $\sum_{t' \in \mathcal{T}(m')} p_\theta(t'|t) = p_\theta(m'|t)$, Eq. 18 can be rewritten as

$$\begin{aligned} \log p(m'|m) &= \log \Big(\sum_{t \in \mathcal{T}(m)} p_\theta(m'|t) p_\phi(t|m) \Big) \\ &= \log \Big(\mathbb{E}_{t \sim p_\phi(t|m)} [p_\theta(m'|t)] \Big) \end{aligned} \tag{19}$$

where

$$p_\theta(m'|t) = \prod_{r \in \mathbb{R}[t]} p_\theta(r) \tag{20}$$

where $\mathbb{R}[t]$ denotes a set of rules in the source tree t and $p_\theta(r)$ is the probability of rule r. Finally, the loss function is defined as an expected negative log-likelihood,

$$L(\theta, \phi) = - \mathbb{E}_{t \sim p_\phi(t|m)} [\log p_\theta(m'|t)] \tag{21}$$

In this work, unlike the original model in [19], the regularization terms are also added in the loss function to ensure the explainability of meta-paths as follows:

$$L(\theta, \phi) = -\mathbb{E}_{t \sim p_\phi(t|m)} \big[\log p_\theta(m'|t) \big] - \lambda \big(\mathrm{R}(m') + \mathrm{C}(m') + \mathrm{D}(m') \big)$$

where $\mathrm{R}(m')$, $\mathrm{C}(m')$ and $\mathrm{D}(m')$ denote the readability, credibility, and diversity of the target meta-path m' respectively, and λ is a parameter for tuning these regularization terms.

Figure 2 illustrates the whole proposed framework. This figure shows an example of the process of translating *UPBPCP* to *UPCP* based on $\mathbb{G}[t]$ and the node alignment between the parse tree t of *UPBPCP* and the parse tree t' of *UPCP*. As shown in this figure, nodes in t' are transduced by certain nodes in t. For instance, node $\mathsf{A}[\alpha_{10}]$ in t' is transduced by α_{10} in t. The grammar rule $\mathsf{A}[\alpha_{10}] \rightarrow \mathsf{B}[\alpha_9]\mathsf{D}[\alpha_5]$ is learned by the decoder. This means that, in the target tree t', $\mathsf{A}[\alpha_{10}]$ is parsed/divided into $\mathsf{B}[\alpha_9]$ which is transduced by α_9 and $\mathsf{D}[\alpha_5]$ which is transduced by α_5.

Table 1. The statistics of MovieLens and Amazon datasets

Dataset	Node type	#nodes	Relation type	#relations
MovieLens	user (U)	1,132	R_{UP}	20,255
	item (P)	3,767	R_{PG}	8,861
	genre (G)	19	R_{PA}	97,791
	actor (A)	53,472	R_{PD}	3,756
	director (D)	1,672	R_{PT}	43,265
	tag (T)	5,209	R_{UV}	1,126
	visual factor (V)	100	R_{PV}	3,121
Amazon	user (U)	39,387	R_{UP}	214,696
	item (P)	23,030	R_{PC}	154,833
	category (C)	1,193	R_{PB}	3,942
	brand (B)	1,181	R_{PH}	65,514
	bought together (H)	25,207	R_{UV}	39,387
	visual factor (V)	100	R_{PV}	23,033

Training. The model parameters are updated by using the gradient descent algorithm. The gradient with respect to θ is computed by using an unbiased Monte Carlo method and backpropagated through the usual inside algorithm [3]. For the gradient with respect to ϕ, the score function estimator with a self-critical baseline [29] is applied as follows:

$$\nabla_{\phi} L(\theta, \phi) \approx \ -(g(\tilde{t}) - g(\hat{t}))\nabla_{\phi} \log p_{\phi}(\tilde{t}|m) \tag{22}$$

where $g(t) = \log p_{\theta}(m'|t)$, $\tilde{t}$ is a sample from $p_{\phi}(t|m)$ and $\hat{t} = \text{argmax}_t p_{\phi}(t, m)$ is the MAP tree of $p_{\phi}(t|m)$.

Inference. Firstly, $\hat{t} = \text{argmax}_t p_{\phi}(t, m)$ is obtained from the parser. Then, K target trees are sampled from $p_{\theta}(t'|\hat{t})$. Among these sampled trees, top-N most-common target trees $t'_1, ..., t'_N$ are selected. Finally, meta-paths $m'_1, ..., m'_N$ corresponding with $t'_1, ..., t'_N$ are treated as final outputs.

5 EXPERIMENTS

5.1 Dataset Generation

To train the proposed meta-path translation model, we first generate two datasets for meta-path translation. To generate a meta-path translation dataset, there are three prerequisites as inputs, i.e., a recommendation dataset, a meta-path based recommendation model, and a set of meta-paths $\mathcal{M}$ for being applied to the selected recommendation model on the selected dataset. Given these inputs, the process of generation is summarized as follows:

(1) For each $m \in \mathcal{M}$, the selected recommendation model is trained based on m on the selected recommendation dataset. Then, each model based on each meta-path is used to compute a set of recommendations. This results in multiple sets of recommendations in which each set contains recommendations based on each meta-path.
(2) With all sets of recommendations based on each and every $m \in \mathcal{M}$, the set of all comparable explainable meta-paths of each m, $f(m)$, is identified. This is done by iteratively considering all possible pairs of meta-paths and validating them based on Definition 5.

(3) For each comparable explainable meta-path m' of m, (m, m') is treated as a sample and added to the dataset for meta-path translation where m is an input (source meta-path) and m' is an output (target meta-path).

In our experiments, the recommendation model in [24] was adopted to predict recommendations based on each meta-path in Step (1). All the parameter settings were the same as in the original paper. This approach uses metapath2vec to find user and item node embeddings and use these embeddings in a CF-KNN model. Note that any meta-path based recommendation approaches are applicable. We selected this approach since it is easy to implement and requires less computational resources than most approaches. Two real-world recommendation datasets were used:

- **MovieLens dataset**[1] [4], an extension of the MovieLens dataset called HetRec2011-MovieLens-2K. It contains user tagging history and movie metadata including genres, actors, and directors. Apart from metadata, we also consider visual factors from [24]. These visual factors represent significant image features extracted from movie poster images[2].
- **Amazon dataset**[3] [16], an e-commerce dataset consisting of users' review history and item metadata divided into various subsets. In this work, we only selected the "Clothing" item subset for the experiments. We only retained 5-rated reviews to ensure the users' satisfaction for learning their preferences. The ratings are converted to implicit feedback to be used in our proposed model. Also, visual factors from [24] are considered. These visual factors were generated from item images provided in this dataset.

To reduce the sparsity, those users who have less than two items and those items that have been interacted with by less than two users were removed. The dataset statistics are summarized in Table 1. For **MovieLens**, there are 7 node types and 12 relation types (inverse relation types included). For **Amazon** dataset, there are 6 node types and 10 relation types (inverse relation types included). For simplicity, in this work, all relations have the same weight which is 1, i.e., $w(r_{x,y}) = 1$ for all $r_{x,y} \in \mathcal{R}$. Assigning different weights for different relations will be considered in future work. For each dataset, meta-paths based on the node and relation types in Table 1 were selected. We used meta-paths that start with the user node type U and end with the item node type P. Based on this condition, the shortest possible meta-path is UP with a length of 1. However, using meta-path UP is equivalent to using only user-item interactions which ignore multi-hop relations. Therefore, we discarded this meta-path and started the range of meta-paths at 2. To determine the maximum length, we considered the fact that increasing the maximum length would result in a significantly larger number of meta-paths. For each and every meta-path, the recommendation model based on this meta-path has to be trained to determine the performance similarities among different meta-paths. A large amount of time would be needed to train all of these models. Therefore, to maintain the feasibility of our experiments while considering the exponential growth of possible meta-paths with increasing length, we set the maximum length of the selected meta-paths to 8. As a result, the lengths of the chosen meta-paths are varied within the range of 2 to 8. In total, Movielens dataset consists of $1,025$ meta-paths, while the Amazon dataset consists of 703 meta-paths. For both datasets, the number of meta-paths of each length can be found in Figure 3.

For Step (2), to identify explainable meta-paths, the maximum length of comparable explainable meta-paths (k), the readability threshold (δ_R), the credibility threshold (δ_C) and the diversity threshold (δ_D) have to be determined. We varied k from 2 to 7 and δ_R, δ_C, and δ_D among $\{0, 0.25, 0.5, 0.75, 1\}$ to examine the number of explainable meta-paths obtained from each combination. Note that since the relation weights of HINs of both datasets were set identically to 1, the credibility was then

[1]https://grouplens.org, http://www.rottentomatoes.com, http://www.imdb.com
[2]http://www.omdbapi.com
[3]http://jmcauley.ucsd.edu/data/amazon/

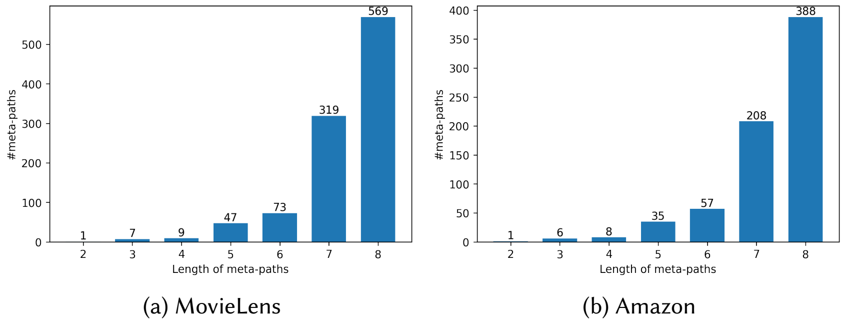


Fig. 3. The number of meta-paths of each length ranging from 2 to 8 used in our experiments on (a) MovieLens and (b) Amazon datasets

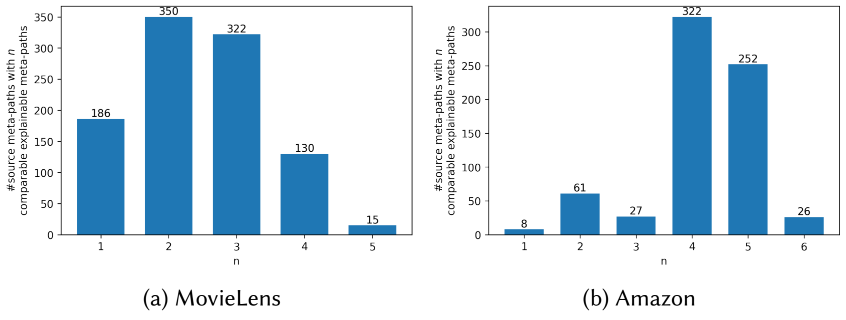


Fig. 4. The number of source meta-paths with n comparable explainable meta-paths in our meta-path translation datasets based on (a) MovieLens and (b) Amazon datasets

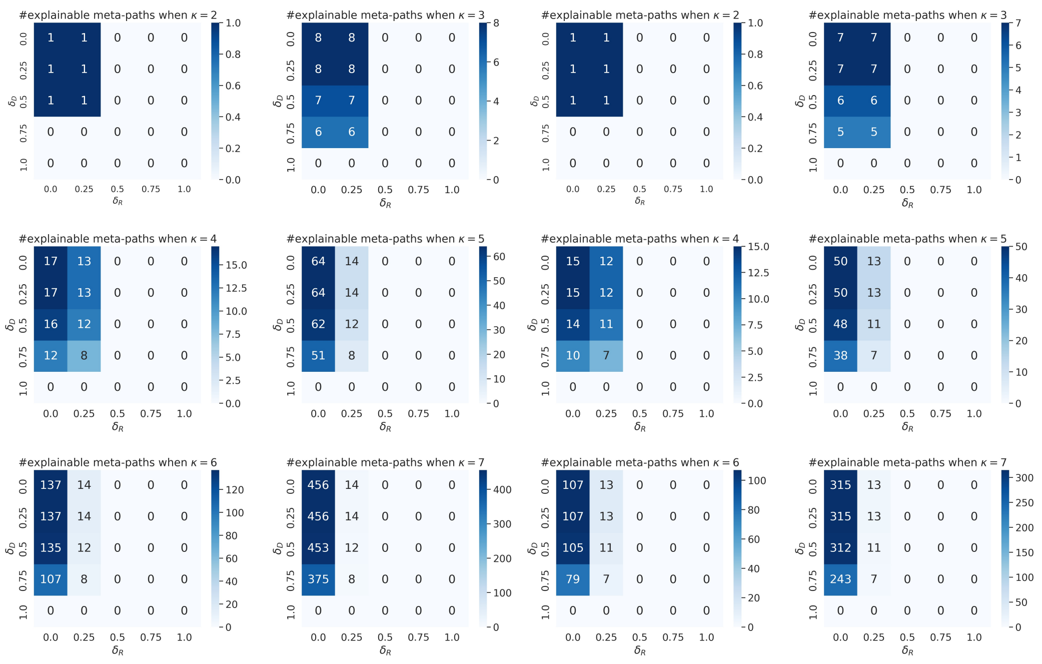
Table 2. Examples of source meta-paths and their corresponding target meta-paths (i.e., their comparable explainable meta-paths) on each dataset

Dataset	Source meta-path	Target meta-path(s)
Movielens	$UPVPVPAP$	$UVUP$
	$UPDPVPTP$	$UPUP$, $UPVP$
	$UVUPTPUVP$	$UPVP$, $UVUP$
Amazon	$UPUPCPHP$	$UPCP$, $UPUP$
	$UPHPBPHP$	$UPBP$, $UPUP$
	$UVPHPBPHP$	UVP, $UVUP$, $UPBP$, $UPVP$

identical for every meta-path. This means that different values of δ_C did not affect the number of explainable meta-paths. Therefore, we only considered the outcomes when varying k, δ_R, and δ_D. Figure 5 shows the number of explainable meta-paths when different k, δ_R, and δ_D were used on **MovieLens** and **Amazon** datasets. To ensure explainability, k should be as short as possible. However, when $k = 2$, there is only one explainable meta-path available for both datasets. Therefore, we selected $k = 3$, which is the second shortest length for both datasets. Then, to include as many explainable meta-paths as possible, we selected the highest thresholds that would allow the

Table 3. The statistics of our meta-path translation datasets generated based on MovieLens and Amazon datasets

Dataset	#total source meta-paths	#total target meta-paths	Training set			Test set		
			#samples	#source meta-paths	#target meta-paths	#samples	#source meta-paths	#target meta-paths
MovieLens	1,003	8	1,738	698	8	709	305	8
Amazon	696	7	2,038	487	7	874	209	7



(a) MovieLens (b) Amazon

Fig. 5. The number of explainable meta-paths when different k, δ_R, and δ_D were used

highest number of explainable meta-paths when $k = 3$, i.e., $\delta_R = 0.25$, and $\delta_D = 0.25$. As a result, the explainable meta-paths for **MovieLens** dataset are UVP, $UPUP$, $UPGP$, $UPAP$, $UPDP$, $UPTP$, $UPVP$ and $UVUP$. For **Amazon** dataset, the explainable meta-paths are UVP, $UPUP$, $UPCP$, $UPBP$, $UPHP$, $UPVP$ and $UVUP$. Comparing these explainable meta-paths with the meta-paths used in the literature [6, 15, 17, 18, 21, 22, 24, 42, 44, 48, 53], we found that they are corresponding with the majority of the shortest meta-paths that were commonly used. Intuitively, shorter meta-paths are easier to comprehend. Since our chosen meta-paths are the shortest meta-paths commonly used, this observation suggests that they are practical and explainable. From Definition 5, at least one performance evaluation metric is required to identify comparable explainable meta-paths. In our experiments, Mean Average Precision@100 (MAP@100) and Mean Recall@100 (Recall@100) were selected to compare performance evaluation values. The last condition in Definition 5 then can be specified as follows: $|\mathrm{A_p}(m) - \mathrm{A_p}(m')| < \delta_1$ or $|\mathrm{A_r}(m) - \mathrm{A_r}(m')| < \delta_2$ where $\mathrm{A_p}(m)$ and $\mathrm{A_p}(m')$ are the MAP@N values of the recommendations based on m and m' respectively, δ_1 is the pre-defined precision threshold, $\mathrm{A_r}(m)$ and $\mathrm{A_r}(m')$ are the Recall@N values of the recommendations based on m and m' respectively and δ_2 is the pre-defined recall threshold. To avoid including or excluding too many comparable explainable meta-paths, the parameters δ_1 and δ_2 were determined by the mean of $|\mathrm{A_p}(m) - \mathrm{A_p}(m')|$ and the mean of $|\mathrm{A_r}(m) - \mathrm{A_r}(m')|$. As a result, δ_1 and δ_2 were set to 0.0001 and 0.01 respectively.

Lastly, in Step (3), each sample was created as aforementioned. In total, there are $1,003$ source meta-paths in **MovieLens** dataset and 696 source meta-paths in **Amazon** dataset. For clarification, these datasets are meta-path translation datasets, not recommendation datasets. For both datasets, the length of source meta-paths ranges from 4 to 8 while the length of target meta-paths ranges from 2 to 3. Each source meta-path corresponds to one or more target meta-paths. The number of source meta-paths with a different number of target meta-paths is shown in Figure 4. On average, one source meta-path corresponds to 2.44 target meta-paths in **MovieLens** dataset and 4.25 target meta-paths in **Amazon** dataset. Table 2 shows some examples of source meta-paths (long and complicated meta-paths in this work) and their corresponding target meta-paths (i.e., their comparable explainable meta-paths). For each dataset, we split 70% of source meta-paths for training and 30% of them for testing. The statistics of our generated datasets are summarized in Table 3. It should be noted that there are certain meta-paths for which no comparable explainable meta-path exists that satisfies the given conditions. Thus, the number of source meta-paths in our meta-path translation datasets is less than the number of all possible meta-paths used for predicting recommendations. The datasets and the implementation of our meta-path translation model can be found in this link[4] with an access request required.

5.2 Training Meta-Path Translation Model

As for the proposed meta-path translation model, two deep learning architectures, i.e., CNN [12] and Transformer, [37] were adopted for latent feature extraction. For training, λ was set to 0.5. The weight decay rate was varied among $\{10^{-3}, 10^{-5}, 10^{-7}, 10^{-10}\}$. The weight decay 10^{-5} and 10^{-7} were selected for **MT-CNN** and **MT-TF** on **MovieLens** dataset respectively. Meanwhile, the weight decay 10^{-5} and 10^{-3} were selected for **MT-CNN** and **MT-TF** on **Amazon** dataset respectively. For sampling the target trees, K was varied among $\{10, 30, 50, 70, 100\}$, and $K = 50$ was used for both models on both datasets.

Baselines. Most state-of-the-art machine translation models heavily relied on pre-training on large corpora [27, 39]. Since pre-training is not applicable in our case, we only compared the proposed meta-path translation model with some Seq2Seq baselines that do not require pre-training as follows:

- **LSTM**[5] [7]: a simple Seq2Seq model based on an LSTM network. Both encoder and decoder embedding sizes were set to 16. The model was trained for 50 epochs. The other parameters were set as in the original model.
- **RLSTM**[6] [34]: a Seq2Seq model based on an LSTM network using reverse-order tokens as input. Both encoder and decoder embedding sizes were set to 16. The model was trained for 50 epochs. The other hyperparameters were set as in the original model.

[4]https://bit.ly/meta-path-translation-model
[5]https://github.com/bentrevett/pytorch-seq2seq
[6]https://pypi.org/project/pytorch-beam-search

- **TF**[7] [37]: a Seq2Seq model based on a standard Transformer model. Both encoder and decoder embedding sizes were set to 16. The model was trained for 100 epochs. The other hyperparameters were set as in the original model.
- **NQCFG**[8] [19]: the original latent neural grammar model based on QCFG without considering the latent feature embedding and the positional embeddings. The embedding size was set to 16 as the other baselines. The rest of the hyperparameters were the same as in the original paper. The model was trained for 50 epochs.
- **MT-CNN**: our proposed meta-path translation model using a CNN[9] [12] for latent feature extraction. As for the CNN module, 2 convolutional layers were used as hidden layers. The number of input and output channels were set to 16 and 32 respectively with the kernel size 3 and the dropout rate 0.25. The hidden output from the last convolutional layer was passed through a linear transformation to obtain a final latent feature embedding. The size of token, positional, and latent feature embeddings was set to 16. The regularization parameter λ was set to 0.9. The model was trained for 50 epochs. The number of samples $K = 50$ was used for inference.
- **MT-TF**: our proposed meta-path translation model using Transformer[10] [37] for latent feature extraction. Two hidden layers with two attention heads were used in the Transformer module. The size of the hidden layer was 16 with a dropout rate of 0.2. All other hyperparameters were as in **MT-CNN**.

During the inference step of **LSTM**, **RLSTM** and **TF**, a beam search strategy [34] was adopted to generate top-N predictions. In this way, it is possible to compare the performances of predicting a set of comparable explainable meta-paths.

Evaluation. Two evaluation metrics, Mean Hit Ratio@N (HR@N) and Mean Recall@N (Recall@N), were used for the comparison of accuracy. Also, Mean Readability@N (RD@N) and Mean Diversity@N (DV@N) were used to evaluate the explainability of the translated meta-paths. These metrics can be computed as follows:

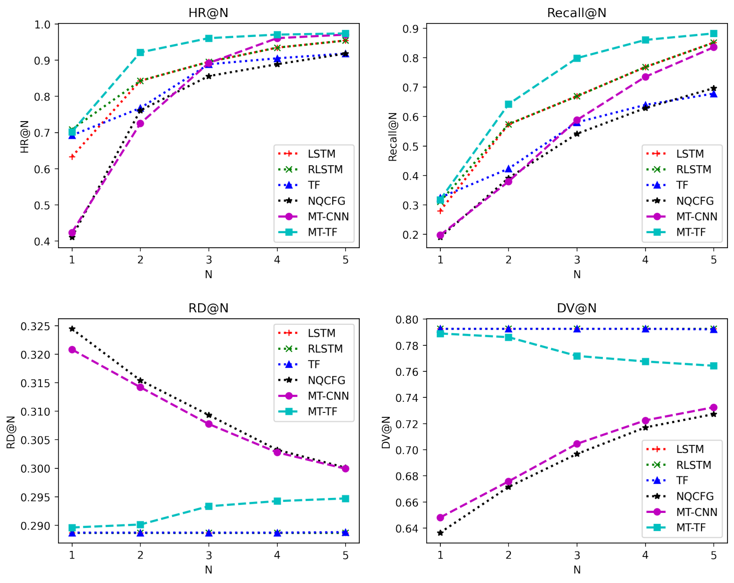
$$RD@N = \sum_{m \in \mathcal{M}^N} \frac{\mathrm{R}(m)}{|\mathcal{M}^N|} \quad \text{and} \quad DV@N = \sum_{m \in \mathcal{M}^N} \frac{\mathrm{D}(m)}{|\mathcal{M}^N|} \tag{23}$$

where $\mathrm{R}(m)$ and $\mathrm{D}(m)$ denote the readability and diversity of m defined in Eq. 1 and 3 and $\mathcal{M}^N$ denote the set of top-N predicted meta-paths. Since, in this work, all relation weights are identical, the credibility of the translated meta-path was not considered. The use of varied relation weights and the result regarding credibility be investigated further in future work.

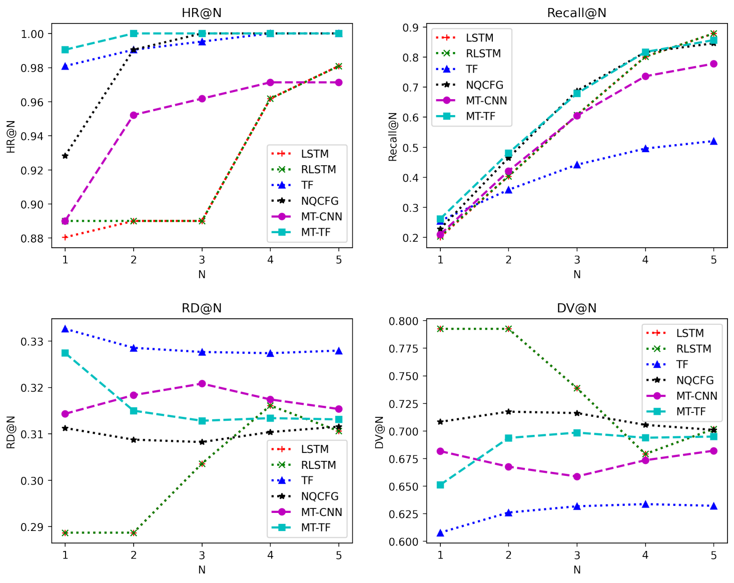
5.3 Performance Evaluation

Figure 6a and Figure 6b show the results on **MovieLens** dataset and **Amazon** dataset respectively. To validate the results, a two-tailed paired sample t-test was conducted with $\alpha = 0.05$. The HR@N and Recall@N improvements of **MT-CNN** and **MT-TF** over the baselines on **MovieLens** dataset and **Amazon** dataset can be found in Table 4 and Table 5 in Appendix A respectively. Overall, on both datasets, our model **MT-TF** performed better than **MT-CNN** and the other baselines including **TF**. On **MovieLens** dataset, **LSTM** and **RLSTM** performed similarly while they both outperformed **TF**. The reason could be that **LSTM** and **RLSTM** can better capture short dependency in short sequences compared to **TF**. **TF** performed well when $N = 1$ which means it effectively

[7]https://pypi.org/project/pytorch-beam-search
[8]https://github.com/yoonkim/neural-qcfg
[9]https://github.com/bentrevett/pytorch-seq2seq
[10]https://github.com/guocheng2018/Transformer-Encoder



(a) MovieLens



(b) Amazon

Fig. 6. Comparison of HR@N, Recall@N, RD@N and DV@N of the proposed approaches and other baselines

predicted a single target meta-path given a source meta-path. It failed to predict a group of target meta-paths since it performed worse than most baselines including our models when N increased. **NQCFG** performed similarly to **TF** except when $N = 1$. Considering our model, **MT-CNN** generally performed worse than both **LSTM** and **RLSTM**. However, it significantly outperformed **TF** and

NQCFG when $N = 4$ and 5. This indicates that the latent features extracted by **MT-CNN** may not be effective on **MovieLens** dataset. Meanwhile, **MT-TF** significantly outperformed **LSTM** and **RLSTM** in terms of both HR@N and Recall@N for every N except when $N = 1$. **MT-TF** also significantly outperformed **TF** when $N = 3, 4$ and 5. This suggests that **LSTM**, **RLSTM**, and **TF** are effective at predicting correct target meta-paths in top-1 and top-2 predictions but they failed to predict a group of correct target meta-paths when N increased on this dataset. **MT-TF** clearly outperformed **NQCFG** in terms of both HR@N and Recall@N. This suggests that adding the positional embeddings and the latent features extracted from a CNN and Transformer improved the performance of meta-path translation. Comparing **MT-TF** and **MT-CNN**, **MT-TF** performed better than **MT-CNN** in terms of both HR@N and Recall@N for every N. This suggests that using Transformer is more effective than a CNN for extracting latent features in the proposed model for **MovieLens** dataset.

On **Amazon** dataset, **LSTM** and **RLSTM** performed similarly to each other. They outperformed **TF** which is a state-of-the-art model in terms of Recall@N. However, they performed worse than all of the baselines in terms of HR@N. On the other hand, **TF** performed particularly well in terms of HR@N but performed worse than the other models including both of our models in terms of Recall@N. This indicates that **TF** is highly effective in predicting one of the correct target meta-paths given a source meta-path but not effective in predicting a group of target meta-paths. **NQCFG** performed similarly to **MT-TF** in terms of Recall@N. However, in terms of HR@N, **MT-TF** significantly outperformed **NQCFG** when $N = 1$. This indicates the effectiveness of including positional embeddings and latent embeddings to correctly predict the correct target meta-path given a source meta-path on the first try. **MT-CNN** did not perform well on this dataset. It performed worse than **NQCFG** in terms of both HR@N and Recall@N for every N. It significantly outperformed **LSTM** and **RLSTM** only in terms of HR@2 and HR@3 and outperformed **TF** only in terms of Recall@N when $N = 3, 4$ and 5. On the other hand, **MT-TF** significantly performed better than both **LSTM** and **RLSTM** in terms of HR@N for every N and Recall@N when $N = 1, 2$ and 3. This suggests that, given a source meta-path, **MT-TF** can predict one of its corresponding target meta-paths more effectively than these two models. **MT-TF** also performed better than **TF** in terms of Recall@N when $N = 2, 3, 4$ and 5. This indicates that **MT-TF** is better than **TF** when predicting a group of target meta-paths. Comparing our two models, **MT-TF** performed better than **MT-CNN** in terms of both HR@N and Recall@N. This suggests that using Transformer to extract latent features for meta-path translation is more effective than using a CNN model which corresponds with the results on **MovieLens** dataset.

In summary, on both datasets, our model **MT-TF** predicted the target meta-paths more effectively than **MT-CNN** and the other baselines including **TF**, which is a state-of-the-art model. This shows the effectiveness of the proposed model. **MT-TF** also outperformed **NQCFG**. This demonstrates the improved performance of the latent neural grammar model using the positional embeddings and the latent features from a CNN and Transformer.

5.4 Readability and Diversity Evaluation

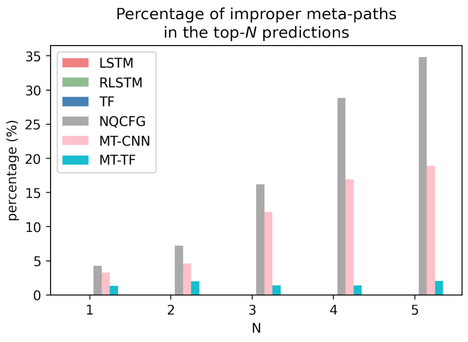
The results of RD@N and DV@N on **MovieLens** dataset and **Amazon** dataset are shown in Figure 6a and Figure 6b respectively. A two-tailed paired sample t-test was conducted with $\alpha = 0.05$. The RD@N and DV@N improvements of **MT-CNN** and **MT-TF** over the baselines on **MovieLens** dataset and **Amazon** dataset can be found in Table 6 and Table 7 in Appendix A respectively. Overall, on both datasets, **MT-CNN** produced target meta-paths with low Diversity but high Readability while **MT-TF** produced target meta-paths with high Diversity but low Readability. On **MovieLens** dataset, **LSTM**, **RLSTM** and **TF** performed similarly. They underperformed the others in terms of RD@N but outperformed the others in terms of DV@N. On the contrary, **NQCFG** performed best in terms of RD@N but it performed worse than the others in terms of DV@N. Comparing our models with the baselines, from Table 6, our model **MT-CNN** performed similarly to **NQCFG** with no significant differences in terms of both RD@N and DV@N. It significantly outperformed the other baselines in terms of RD@N but performed worse than them in terms of DV@N. This suggests that **MT-CNN** tends to predict shorter target meta-paths with high Readability but low Diversity as **NQCFG**. **MT-TF** performed worse than **NQCFG** while slightly outperformed **LSTM**, **RLSTM**, and **TF** in terms of RD@N. In terms of DV@N, **MT-TF** only performed better than **NQCFG** while performing slightly worse than **LSTM**, **RLSTM**, and **TF**. Overall, the results indicate that those models based on LSTM and Transformer networks predicted the target meta-paths with lower Readability but higher Diversity compared to the others. Meanwhile, **NQCFG** and **MT-CNN** predicted the target meta-paths with higher Readability but lower Diversity than the others. This suggests that **NQCFG** and **MT-CNN** prioritized predicting shorter meta-paths with less number of node and relation types compared to the others. Comparing our models, **MT-CNN** performed better than **MT-TF** in terms of Readability but the result is the opposite in terms of Diversity on this dataset.

On **Amazon** dataset, **LSTM** and **RLSTM** performed similarly in terms of both RD@N and DV@N. They produced target meta-paths with high Diversity but low Readability. Their results on this dataset are similar to those of **MovieLens** dataset. They show that both **LSTM** and **RLSTM** prioritized Diversity over Readability for meta-path translation. **TF**, on the other hand, predicted target meta-paths with higher Readability but lower Diversity compared to **LSTM** and **RLSTM**. This is opposite to the result on **MovieLens** dataset. One possible reason is that **Amazon** dataset contains fewer node and relation types than **MovieLens** dataset. This results in less diverse meta-paths in the training set. As a result, this training set may not be sufficient for **TF** to learn the diversity of node and relation types in meta-path translation. Thus, it failed to predict target meta-paths with high Diversity on **Amazon** dataset. **NQCFG** underperformed both of our models in terms of RD@N but outperformed them in terms of DV@N. Table 7 shows that **MT-CNN** significantly outperformed **LSTM** and **RLSTM** in terms of RD@N for every N except $N = 4$ and **NQCFG** in terms of RD@N for every N except $N = 1$. Similarly, **MT-TF** significantly outperformed **LSTM** and **RLSTM** in terms of RD@N when $N = 1, 2$ and 3. and also **NQCFG** in terms of RD@N for every N except $N = 5$. This indicates the effectiveness of **MT-CNN** and **MT-TF** in predicting target meta-paths with high Readability compared to **LSTM**, **RLSTM**, and **NQCFG**. However, in terms of DV@N, both **MT-CNN** and **MT-TF** performed worse than most of the baselines except **TF**. Comparing our two models, **MT-CNN** performed better than **MT-TF** in terms of RD@N. Conversely, **MT-TF** performed better than **MT-CNN** in terms of DV@N.

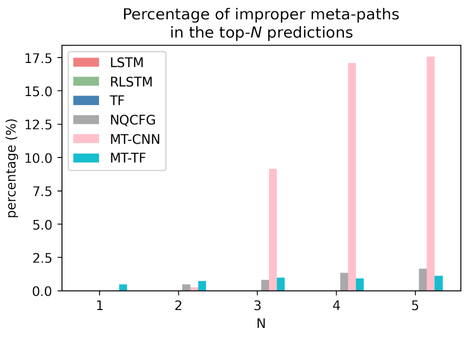
Overall, on both datasets, it can be seen that **MT-CNN** focused on predicting short target meta-paths with low Diversity to achieve high Readability. Meanwhile, **MT-TF** focused on predicting target meta-paths consisting of various relation types resulting in high Diversity. Considering all HR@N, Recall@N, RD@N, and DV@N results, it can be inferred that **MT-TF** exhibits greater accuracy and the ability to predict target meta-paths with higher Diversity compared to MT-CNN. However, if the priority is Readability, **MT-CNN** is more suitable than **MT-TF**.

5.5 Error Analysis

This section examines the meta-path error predicted by the proposed models and the other baselines. Besides being comparable explainable meta-paths, predicted target meta-paths must hold two properties: (1) they must start with the user node type (U) and end with the item node type (P), i.e., they should follow $UN_1N_2, ..., N_lP$ and (2) they must consist of existing relation type(s) defined in the graph, i.e., given a graph schema $\mathbb{G} = (\mathbb{N}, \mathbb{R}, \mathbb{W})$, for any target meta-path $UN_1N_2, ..., N_lP$,



(a) MovieLens



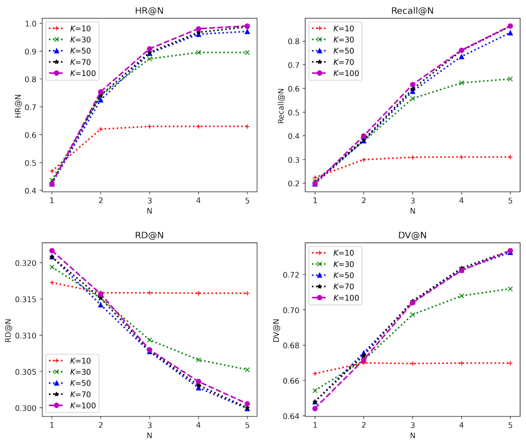
(b) Amazon

Fig. 7. Comparisons of the percentages of improper meta-paths in the top-N predictions obtained from the proposed approaches and other baselines

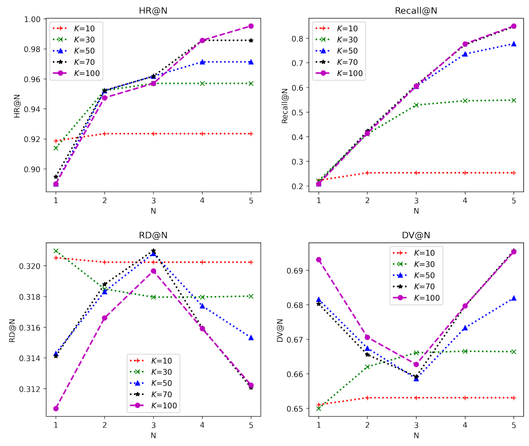
$R_{UN_1}, R_{N_1,N_2}, ..., R_{N_l,P} \in \mathbb{R}$. A meta-path that holds these properties is referred to as *a proper meta-path*. Meanwhile, a meta-path that does not hold these properties is considered *an improper meta-path*. The percentage of improper meta-paths in the top-N predictions obtained from each model was computed to examine the error of predicting improper target meta-paths. The results are shown in Figure 7. From this figure, there is no improper meta-path predicted by **LSTM**, **RLSTM**, and **TF** on both **MovieLens** and **Amazon** datasets for every N. This demonstrates their effectiveness in modeling dependences between the first and the last node types and also two consecutive node types in the target meta-paths. For **MovieLens**, our models **MT-CNN** and **MT-TF** predicted more improper meta-paths than **LSTM**, **RLSTM**, and **TF**. However, both **MT-CNN** and **MT-TF** predicted substantially fewer improper meta-paths than **NQCFG** which is the original latent neural grammar model without using the latent feature and positional embeddings. This indicates the effectiveness of using the latent feature embeddings from both CNN and Transformer models and also the positional embeddings. The latent feature embeddings from both CNN and Transformer models play the role of capturing linear dependencies between node types in the meta-paths to avoid predicting nonexistent relation types. Meanwhile, the positional embeddings distinguish the starting and ending node types from the others which helps the model to predict the first and the last node types in the target meta-paths. In the case of **Amazon** dataset, **NQCFG** and **MT-TF** performed similarly. **MT-TF** predicted slightly more improper meta-paths than **NQCFG** when $N = 1, 2$ and 3. However, **MT-CNN** failed to predict proper target meta-paths. One possible reason is that the number of training samples in **Amazon** dataset is less than the number of training samples in **MovieLens** dataset. This may not be sufficient for **MT-CNN** to capture the dependencies between node types effectively. On both datasets, **MT-TF** predicted improper meta-paths less than 5% of the predicted meta-paths for every N which is less than **MT-CNN**. This suggests that **MT-TF** is more effective than **MT-CNN** in predicting proper target meta-paths. This corresponds to the accuracy results in Section 5.3.

5.6 Hyperparameter Analysis

In the following sections, we explore the analysis of hyperparameters to examine their effects on the performance of the proposed models. The examination focuses on two hyperparameters, i.e., the number of sampled target trees (K) and weight decay (w_d).

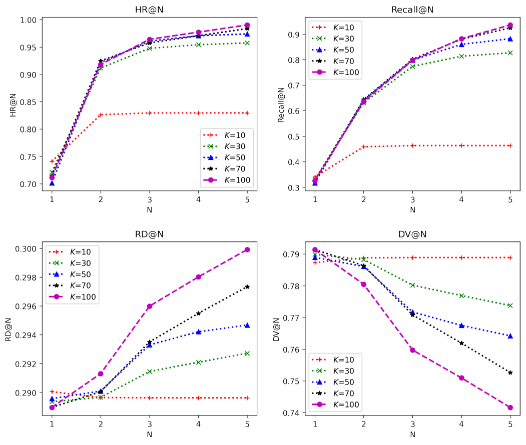


(a) MovieLens

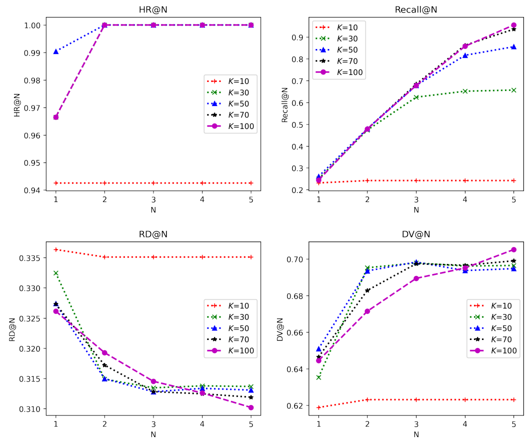


(b) Amazon

Fig. 8. Comparison of HR@N and Recall@N when using different K for sampling the target trees in **MT-CNN**



(a) MovieLens



(b) Amazon

Fig. 9. Comparison of HR@N and Recall@N when using different K for sampling the target trees in **MT-TF**

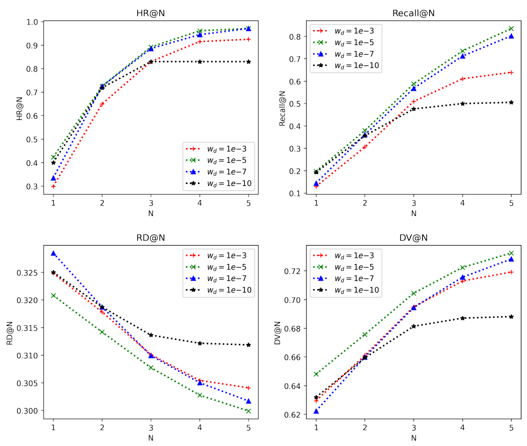
5.6.1 *Effect of the number of sampled target trees (K).*

In this section, the effect of the parameter K (the number of sampled target trees from the decoder) is discussed. This parameter was varied among 10, 30, 50, 70, and 100 to examine the differences in HR@N, Recall@N, RD@N, and DV@N values. The results of **MT-CNN** when varying K on both datasets are shown in Figure 8. From this figure, the higher values of K generally lead to better HR@N and Recall@N on both datasets, as they increase with an increase in K for every value of N. However, when comparing $K = 50$, 70, and 100, the differences in HR@N and Recall@N are almost negligible. This suggests that using $K = 50$ is sufficient to achieve accurate outcomes. Regarding Readability and Diversity on **MovieLens** dataset, RD@N of the predicted target meta-paths by **MT-CNN** decreases as K increases. Conversely, DV@N of these meta-paths increases with an increase in K. On **Amazon** dataset, RD@N generally decreases as K increases, following a similar pattern as observed on **MovieLens** dataset. However, there is some inconsistency in this pattern for RD@3. In terms of DV@N, it also tends to increase with an increase in K, except for DV@3. The results obtained from **MT-TF** on both datasets are presented in Figure 9. This figure illustrates that as the value of K increases, **MT-TF** predicts target meta-paths with higher HR@N and Recall@N on both datasets. Regarding Readability, on **MovieLens** dataset, RD@N increases with the increment of K, whereas DV@N decreases. Conversely, on **Amazon** dataset, the results are opposite to those of **MovieLens** dataset. RD@N decreases with the increase in K, while DV@N increases. Overall, for both **MT-CNN** and **MT-TF**, HR@N and Recall@N improve with the increase in K. However, using larger values of K is computationally expensive, and therefore, using $K = 50$ is sufficient as it yields similar results to $K = 100$ in most cases. On the other hand, the trends observed in RD@N and DV@N vary among different models and datasets. Generally, using $K = 50$ achieves moderate results balancing between RD@N and DV@N.

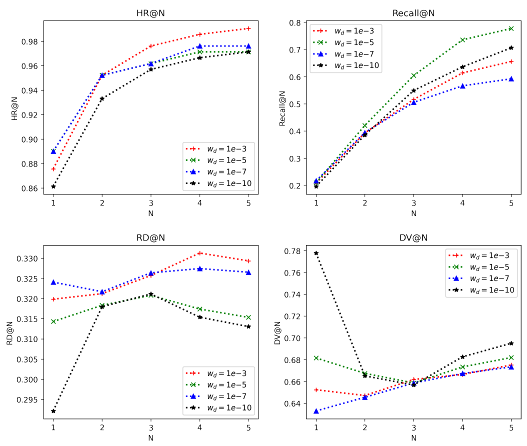
5.6.2 Effect of weight decay (w_d). The effect of weight decay (w_d) when training the proposed models was also examined. We varied weight decay among $\{10^{-3}, 10^{-5}, 10^{-7}, \text{and } 10^{-10}\}$ and examined the results of our models. Figures 10a and 10b show the results of using different values of weight decay in **MT-CNN** on **MovieLens** dataset and **Amazon** dataset respectively. On **MovieLens** dataset, **MT-CNN** using $w_d = 10^{-5}$ achieved the highest HR@N and Recall@N compared to the other models. On **Amazon** dataset, **MT-CNN** using $w_d = 10^{-5}$ achieved only the highest Recall@N compared to the others. In terms of HR@N, **MT-CNN** using $w_d = 10^{-3}$ generally performed better than the others. However, the difference in HR@N between using $w_d = 10^{-3}$ and $w_d = 10^{-5}$ is smaller than the difference in Recall@N between using $w_d = 10^{-3}$ and $w_d = 10^{-5}$. Therefore, $w_d = 10^{-5}$ was chosen for comparison with the baselines on the Amazon dataset. Regarding RD@N and DV@N, on **MovieLens** dataset, **MT-CNN** using $w_d = 10^{-10}$ generally performed better than the others in terms of RD@N while performing worse than the others in terms of DV@N. On the other hand, **MT-CNN** using $w_d = 10^{-5}$ performed best in terms of DV@N but performed worst in terms of RD@N. On **Amazon** dataset, **MT-CNN** using $w_d = 10^{-10}$ gave higher DV@N and lower RD@N than the others. Meanwhile, the model using $w_d = 10^{-7}$ gave the higher RD@N and lower DV@N than the others. The model using $w_d = 10^{-5}$ gave the second-best DV@N and the second-worst RD@N. As for **MT-TF**, the results on **MovieLens** dataset and **Amazon** dataset are shown in Figures 11a and 11b respectively. On **MovieLens** dataset, reducing w_d resulted in higher HR@N and Recall@N until they peaked at $w_d = 10^{-7}$. Conversely, on **Amazon** dataset, reducing w_d resulted in lower HR@N and Recall@N and $w_d = 10^{-3}$ yielded the best performance. In terms of RD@N, on **MovieLens** dataset, **MT-TF** using $w_d = 10^{-10}$ outperformed the others when $N = 1$ while the model using $w_d = 10^{-3}$ outperformed the others for $N = 2, 3, 4$ and 5. The model using $w_d = 10^{-7}$ performed worst compared to the others. On the contrary, in terms of DV@N, the model using $w_d = 10^{-7}$ performed best while the models using $w_d = 10^{-3}$ and $w_d = 10^{-10}$ performed worse than the others. On **Amazon** dataset, **MT-TF** using $w_d = 10^{-5}$ gave the highest DV@N and lowest RD@N compared to the others. On the contrary, the model using $w_d = 10^{-3}$ gave the highest RD@N and lowest DV@N. Overall, the results suggest that using too-high or too-low weight decay may result in low accuracy. On **MovieLens** dataset, using a smaller weight decay generally resulted in better performance. Meanwhile, on **Amazon** dataset, using a higher weight decay resulted in better performance, especially for **MT-TF**.

5.7 Computational Complexity Analysis

Let l and l' be the length of a source meta-path m and a target meta-path m' respectively. For the parser, sampling the tree $\tilde{t}$ and the argmax tree $\hat{t}$ requires $O(l^3)$ by dynamic programming. For the

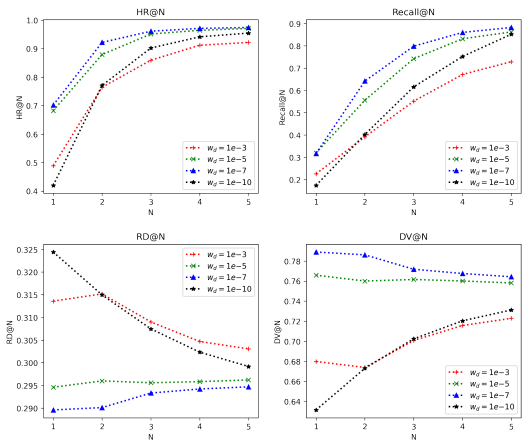


(a) MovieLens

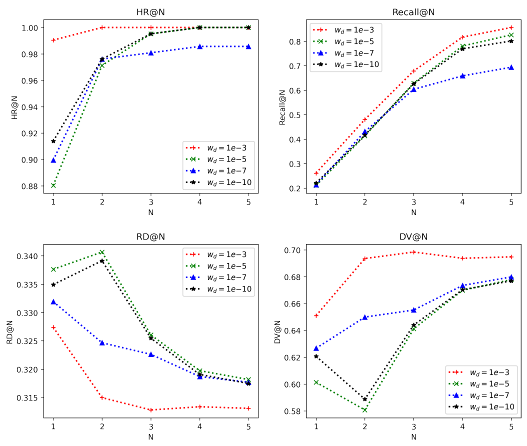


(b) Amazon

Fig. 10. Comparison of HR@N and Recall@N when using different weight decay w_d for training **MT-CNN**



(a) MovieLens



(b) Amazon

Fig. 11. Comparison of HR@N and Recall@N when using different weight decay w_d for training **MT-TF**

encoder, each CNN layer requires $O(l \cdot S \cdot E^2)$ while each self-attention layer of Transformer requires $O(l^2 \cdot E)$ where S is the kernel size of convolutions and E is the size of the embeddings. TreeLSTM requires $O(l)$ to generate the final token representations. For the decoder, by dynamic programming, computing $\log p_\phi(\tilde{t}|m)$ requires $O(l^3)$. Similarly, computing $\log p_\theta(m'|\tilde{t})$ and $\log p_\theta(m'|\hat{t})$ requires $O(l^3 l'^3)$. Thus, the complexity of both **MT-CNN** and **MT-TF** is still dominated by $O(l^3 l'^3)$ as the original latent neural grammar model **NQCFG**. This suggests that using the latent feature extraction module and the positional and latent feature embeddings does not severely affect the overall complexity.

6 CONCLUSIONS

Meta-path based recommendations can be intuitively explained by the meta-paths that are used for predicting these recommendations. However, recommendations based on long and complicated meta-paths can be difficult to explain. In this work, we proposed a method to better explain meta-path based recommendations. First, we introduced the definition of **meta-path explainability** based on meta-path readability, credibility, and diversity. Based on this definition, we proposed how to find **comparable explainable meta-paths** of a given meta-path. These comparable explainable meta-paths can be used as alternative explanations that are easier to understand than the given one. To find comparable explainable meta-paths of any meta-path, the **meta-path translation model** was proposed. Specifically, inspired by QCFG, the meta-path grammar was first introduced. Based on the meta-path grammar, a Seq2Seq model that maps a given meta-path to its comparable explainable ones was proposed. This model leverages both latent features extracted by CNN and Transformer models and hierarchical features extracted by TreeLSTM simultaneously.

Two meta-path translation datasets were generated based on two real-world datasets. In these datasets, each input is a source meta-path and each output is a group of its comparable explainable meta-paths (target meta-paths). These datasets allow us to learn how to explain recommendations of real-world datasets by mapping a source meta-path to target meta-paths. This mapping can be considered as a one-to-many task where one source meta-path can yield multiple target meta-paths. Extensive experiments were conducted on these two generated datasets. The results show that our proposed model performed better than other baselines in terms of both HR@N and Recall@N for both datasets. This indicates the effectiveness of using both latent features extracted from CNN/Transformer and hierarchical features from the TreeLSTM model. Moreover, compared to the baselines, our models predicted a group of target meta-paths with higher HR@N and Recall@N when $N > 1$. Apart from accuracy, the meta-path readability and diversity were also evaluated based on two metrics, Readability@N and Diversity@N. According to the results, the proposed approach using a CNN model performed better in terms of Readability. On the other hand, the proposed approach using a Transformer model produced the target meta-paths with higher Diversity. Both approaches show a capability of maintaining a better trade-off between accuracy and readability/diversity in translating meta-paths. Also, by including the latent feature embeddings and positional embeddings, our models predicted fewer improper target meta-paths compared to the original latent neural grammar model. This indicates the effectiveness of the latent features and positional embeddings.

6.1 Limitations

The limitations of this work lie in the experiments. Generating a dataset for meta-path translation requires training multiple recommender systems for each and every meta-path considered. Consequently, with an increase in the number of meta-paths, the process of generating a new dataset becomes increasingly time-consuming. This leads to the small number of meta-paths in the generated datasets and the limited number of datasets. Another limitation is the baseline comparison. State-of-the-art Seq2Seq models for machine translation rely heavily on pre-training on large text corpora. In contrast, the task of meta-path translation cannot leverage such pre-training as there are no corpora of meta-paths. As a result, the proposed approach was compared with baselines that also do not utilize pre-training on corpora. These baselines may exclude several state-of-the-art models such as Large Language Models (LLMs). In terms of evaluation, we compared the readability and diversity of the generated target meta-paths across different models. This comparison ensures that the target meta-paths produced by our proposed models have higher readability and diversity compared to the baselines. However, we did not validate whether the generated comparable explainable meta-paths are conducive to explaining the recommendations.

6.2 Future Work

For future work, we aim to explore other real-world datasets to generate more datasets for explaining meta-path based recommendations. Assigning different relation weights in graphs to examine the proposed approach performance will also be considered. To improve the meta-path translation model, it is worth exploring other neural network architectures for extracting the latent features from a meta-path. The potential adoption of Large Language Models to enhance meta-path based and HIN-based recommendations will be investigated. Furthermore, human user studies will be conducted in addition to empirical evaluations.

REFERENCES

[1] James S. Adelman, Gordon D.A. Brown, and José F. Quesada. 2006. Contextual Diversity, Not Word Frequency, Determines Word-Naming and Lexical Decision Times. *Psychological Science* 17, 9 (2006), 814–823.

[2] Gediminas Adomavicius and Alexander Tuzhilin. 2005. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE Transactions on Knowledge and Data Engineering* 17, 6 (June 2005), 734–749.

[3] James K Baker. 1979. Trainable grammars for speech recognition. *The Journal of the Acoustical Society of America* 65, S1 (1979), S132–S132.

[4] Iván Cantador, Peter Brusilovsky, and Tsvi Kuflik. 2011. 2nd Workshop on Information Heterogeneity and Fusion in Recommender Systems (HetRec 2011). In *Proceedings of the 5th ACM conference on Recommender systems.*

[5] Yaomin Chang, Chuan Chen, Weibo Hu, Zibin Zheng, Xiaocong Zhou, and Shouzhi Chen. 2022. Megnn: Meta-Path Extracted Graph Neural Network for Heterogeneous Graph Representation Learning. *Knowledge-Based System* 235, C (jan 2022), 11 pages.

[6] Hongxu Chen, Yicong Li, Xiangguo Sun, Guandong Xu, and Hongzhi Yin. 2021. Temporal Meta-path Guided Explainable Recommendation. *Proceedings of the 14th ACM International Conference on Web Search and Data Mining* (2021).

[7] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).* Association for Computational Linguistics, Doha, Qatar, 1724–1734.

[8] Damian Clifford and Jef Ausloos. 2018. Data Protection and the Role of Fairness. *Yearbook of European Law* 37 (08 2018), 130–187.

[9] Yuxiao Dong, Nitesh V Chawla, and Ananthram Swami. 2017. metapath2vec: Scalable Representation Learning for Heterogeneous Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* 135–144.

[10] Heike Felzmann, Eduard Fosch Villaronga, Christoph Lutz, and Aurelia Tamò-Larrieux. 2019. Transparency you can trust: Transparency requirements for artificial intelligence between legal norms and contextual concerns. *Big Data & Society* 6, 1 (2019), 2053951719860542.

[11] Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style Transfer in Text: Exploration and Evaluation. *Proceedings of the AAAI Conference on Artificial Intelligence* 32, 1 (Apr. 2018).

[12] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional Sequence to Sequence Learning. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70* (Sydney, NSW, Australia) *(ICML'17).* JMLR.org, 1243–1252.

[13] Bryce Goodman and Seth Flaxman. 2017. European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation". *AI Magazine* 38, 3 (Oct. 2017), 50–57.

[14] Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A Survey on Knowledge Graph-Based Recommender Systems. *IEEE Transactions on Knowledge & Data Engineering* (2020).

[15] Zhenyu Han, Fengli Xu, Jinghan Shi, Yu Shang, Haorui Ma, Pan Hui, and Yong Li. 2020. Genetic Meta-Structure Search for Recommendation on Heterogeneous Information Network. 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, 455–464.

[16] Ruining He and Julian McAuley. 2016. Ups and Downs: Modeling the Visual Evolution of Fashion Trends with One-Class Collaborative Filtering. In *Proceedings of the 25th International Conference on World Wide Web*. 507–517.

[17] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Tianchi Yang. 2018. Local and Global Information Fusion for Top-N Recommendation in Heterogeneous Information Network. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management* (Torino, Italy) *(CIKM '18)*. Association for Computing Machinery, New York, NY, USA, 1683–1686.

[18] Binbin Hu, Chuan Shi, Wayne Xin Zhao, and Philip S. Yu. 2018. Leveraging Meta-Path Based Context for Top-N Recommendation with A Neural Co-Attention Model. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1531–1540.

[19] Yoon Kim. 2021. Sequence-to-Sequence Learning with Latent Neural Grammars. In *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021)*.

[20] Y. Koren, R. Bell, and C. Volinsky. 2009. Matrix Factorization Techniques for Recommender Systems. *Computer* 42, 8 (2009), 30–37.

[21] Huizhi Liang, Zehao Liu, and Thanet Markchom. 2022. Relation-Aware Blocking for Scalable Recommendation Systems. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management* (Atlanta, GA, USA) *(CIKM '22)*. Association for Computing Machinery, New York, NY, USA, 4214–4218.

[22] Huizhi Liang and Thanet Markchom. 2022. TNE: A General Time-Aware Network Representation Learning Framework for Temporal Applications. *Knowledge-Based Systems* 240, C (mar 2022), 17 pages.

[23] Weizhi Ma, Woojeong Jin, Min Zhang, Chenyang Wang, Yue Cao, Yiqun Liu, Shaoping Ma, and Xiang Ren. 2019. Jointly learning explainable rules for recommendation with knowledge graph. *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019* (2019), 1210–1221.

[24] Thanet Markchom and Huizhi Liang. 2021. Augmenting Visual Information in Knowledge Graphs for Recommendations. In *26th International Conference on Intelligent User Interfaces*. 475–479.

[25] R. Thomas McCoy, Roberta Frank, and Tal Linzen. 2020. Does Syntax Need to Grow on Trees? Sources of Hierarchical Inductive Bias in Sequence-to-Sequence Networks. *Transactions of the Association for Computational Linguistics* 8 (2020), 125–140.

[26] Makbule Gulcin Ozsoy, Diarmuid O'Reilly-Morgan, Panagiotis Symeonidis, Elias Z. Tragos, Neil Hurley, Barry Smyth, and Aonghus Lawlor. 2020. MP4Rec: Explainable and Accurate Top-N Recommendations in Heterogeneous Information Networks. *IEEE Access* (2020).

[27] Prajit Ramachandran, Peter Liu, and Quoc Le. 2017. Unsupervised Pretraining for Sequence to Sequence Learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Copenhagen, Denmark, 383–391.

[28] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian Personalized Ranking from Implicit Feedback. In *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*. 10 pages.

[29] Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-Critical Sequence Training for Image Captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[30] Andrew I. Schein, Alexandrin Popescul, Lyle H. Ungar, and David M. Pennock. 2002. Methods and Metrics for Cold-Start Recommendations. In *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (Tampere, Finland) *(SIGIR '02)*. Association for Computing Machinery, New York, NY, USA, 253–260.

[31] Tian Shi, Yaser Keneshloo, Naren Ramakrishnan, and Chandan K. Reddy. 2021. Neural Abstractive Text Summarization with Sequence-to-Sequence Models. *ACM/IMS Trans. Data Sci.* 2, 1, Article 1 (jan 2021), 37 pages.

[32] David A. Smith and Jason Eisner. 2006. Quasi-Synchronous Grammars: Alignment by Soft Projection of Syntactic Dependencies. In *Proceedings of the Workshop on Statistical Machine Translation* (New York City, New York) *(StatMT '06)*. Association for Computational Linguistics, USA, 23–30.

[33] Yizhou Sun, Jiawei Han, X. Yan, Philip S. Yu, and Tianyi Wu. 2011. PathSim: Meta Path-Based Top-K Similarity Search in Heterogeneous Information Networks. *Proc. VLDB Endow.* 4 (2011), 992–1003.

[34] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2* (Montreal, Canada) *(NIPS'14)*. MIT Press, Cambridge, MA, USA, 3104–3112.

[35] Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved Semantic Representations From Tree-Structured Long Short-Term Memory Networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. Association for Computational Linguistics, Beijing, China, 1556–1566.

[36] Nava Tintarev and Judith Masthoff. 2007. A Survey of Explanations in Recommender Systems. In *2007 IEEE 23rd International Conference on Data Engineering Workshop*. 801–810.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is All you Need. In *Advances in Neural Information Processing Systems*, I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.), Vol. 30. Curran Associates, Inc.

[38] 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.

[39] Wenxuan Wang, Wenxiang Jiao, Yongchang Hao, Xing Wang, Shuming Shi, Zhaopeng Tu, and Michael Lyu. 2022. Understanding and Improving Sequence-to-Sequence Pretraining for Neural Machine Translation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 2591–2600.

[40] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. KGAT: Knowledge Graph Attention Network for Recommendation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 950–958.

[41] Xin Wang, Ying Wang, and Yunzhi Ling. 2020. Attention-Guide Walk Model in Heterogeneous Information Network for Multi-Style Recommendation Explanation. *Proceedings of the AAAI Conference on Artificial Intelligence* 34, 04 (2020), 6275–6282.

[42] Zekai Wang, Hongzhi Liu, Yingpeng Du, Zhonghai Wu, and Xing Zhang. 2019. Unified Embedding Model over Heterogeneous Information Network for Personalized Recommendation. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence* (Macao, China) *(IJCAI'19)*. AAAI Press, 3813–3819.

[43] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-Based Recommendation with Graph Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence* 33, 01 (Jul. 2019), 346–353.

[44] Yikun Xian, Zuohui Fu, S. Muthukrishnan, Gerard de Melo, and Yongfeng Zhang. 2019. Reinforcement Knowledge Graph Reasoning for Explainable Recommendation. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. 285–294.

[45] Fan Yang, Ninghao Liu, Suhang Wang, and Xia Hu. 2018. Towards Interpretation of Recommender Systems with Sorted Explanation Paths. *Proceedings - IEEE International Conference on Data Mining, ICDM* (2018).

[46] Yaming Yang, Ziyu Guan, Jianxin Li, Wei Zhao, Jiangtao Cui, and Quan Wang. 2021. Interpretable and Efficient Heterogeneous Graph Convolutional Network. *IEEE Transactions on Knowledge and Data Engineering* (2021), 1–1.

[47] 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 & data mining*. 974–983.

[48] Xiao Yu, Xiang Ren, Yizhou Sun, Quanquan Gu, Bradley Sturt, Urvashi Khandelwal, Brandon Norick, and Jiawei Han. 2014. Personalized entity recommendation: A heterogeneous information network approach. *WSDM 2014 - Proceedings of the 7th ACM International Conference on Web Search and Data Mining* (2014), 283–292.

[49] Xiao Yu, Xiang Ren, Yizhou Sun, Bradley Sturt, Urvashi Khandelwal, Quanquan Gu, Brandon Norick, and Jiawei Han. 2013. Recommendation in Heterogeneous Information Networks with Implicit User Feedback. In *Proceedings of the 7th ACM Conference on Recommender Systems* (Hong Kong, China) *(RecSys '13)*. Association for Computing Machinery, New York, NY, USA, 347–350.

[50] Seongjun Yun, Minbyul Jeong, Sungdong Yoo, Seunghun Lee, Sean S. Yi, Raehyun Kim, Jaewoo Kang, and Hyunwoo J. Kim. 2022. Graph Transformer Networks: Learning meta-path graphs to improve GNNs. *Neural Networks* 153 (2022), 104–119.

[51] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep Learning Based Recommender System: A Survey and New Perspectives. *Comput. Surveys* 52, 1, Article 5 (Feb. 2019), 38 pages.

[52] Yongfeng Zhang and Xu Chen. 2020. Explainable Recommendation: A Survey and New Perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101.

[53] Huan Zhao, Quanming Yao, Jianda Li, Yangqiu Song, and Dik Lun Lee. 2017. Meta-Graph Based Recommendation Fusion over Heterogeneous Information Networks. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (Halifax, NS, Canada) *(KDD '17)*. Association for Computing Machinery, New York, NY, USA, 635–644.

A PERFORMANCE, READABILITY, AND DIVERSITY IMPROVEMENTS OF THE PROPOSED MODELS OVER THE BASELINES

Table 4. HR@N and Recall@N improvements over the baselines on **MovieLens** dataset. The improvements that are statically significant are in bold. The value indicated with a star (*) is a positive and statistically significant value.

Baseline	MT-CNN									
	Δ HR@1	Δ HR@2	Δ HR@3	Δ HR@4	Δ HR@5	Δ Recall@1	Δ Recall@2	Δ Recall@3	Δ Recall@4	Δ Recall@5
LSTM	**-0.2098**	**-0.1180**	-0.0033	**0.0262***	0.0164	**-0.0825**	**-0.1943**	**-0.0810**	-0.0332	-0.0157
RLSTM	**-0.2852**	**-0.1180**	-0.0033	**0.0262***	0.0164	**-0.1139**	**-0.1943**	**-0.0810**	-0.0332	-0.0157
TF	**-0.2689**	**-0.1967**	-0.0295	**0.0393***	**0.0492***	**-0.1227**	**-0.2399**	**-0.0950**	**0.0492***	**0.1496***
NQCFG	0.0131	-0.0361	0.0361	**0.0721***	**0.0525***	0.0071	-0.0117	**0.0466***	**0.1062***	**0.1387***
Baseline	MT-TF									
	Δ HR@1	Δ HR@2	Δ HR@3	Δ HR@4	Δ HR@5	Δ Recall@1	Δ Recall@2	Δ Recall@3	Δ Recall@4	Δ Recall@5
LSTM	**0.0689***	**0.0787***	**0.0656***	**0.0361***	**0.0197***	0.0377	**0.0683***	**0.1292***	**0.0920***	**0.0313***
RLSTM	-0.0066	**0.0787***	**0.0656***	**0.0361***	**0.0197***	0.0063	**0.0683***	**0.1292***	**0.0920***	**0.0313***
TF	0.0098	0.0000	**0.0393***	**0.0492***	**0.0525***	-0.0025	0.0226	**0.1152***	**0.1744***	**0.1966***
NQCFG	**0.2918***	**0.1607***	**0.1049***	**0.0820***	**0.0557***	**0.1273***	**0.2508***	**0.2568***	**0.2314***	**0.1857***

Table 5. HR@N and Recall@N improvements over the baselines on **Amazon** dataset. The improvements that are statically significant are in bold. The value indicated with a star (*) is a positive and statistically significant value.

Baseline	MT-CNN									
	Δ HR@1	Δ HR@2	Δ HR@3	Δ HR@4	Δ HR@5	Δ Recall@1	Δ Recall@2	Δ Recall@3	Δ Recall@4	Δ Recall@5
LSTM	0.0096	**0.0622***	**0.0718***	0.0096	-0.0096	0.0092	0.0172	-0.0025	**-0.0651**	**-0.1006**
RLSTM	0.0000	**0.0622***	**0.0718***	0.0096	-0.0096	0.0060	0.0172	-0.0025	**-0.0651**	**-0.1006**
TF	**-0.1005**	**-0.0478**	**-0.0383**	**-0.0287**	**-0.0287**	-0.0518	-0.0229	**0.0890***	**0.2128***	**0.2546***
NQCFG	**-0.0383**	**-0.0383**	**-0.0383**	**-0.0287**	**-0.0287**	-0.0187	**-0.0439**	**-0.0830**	**-0.0802**	**-0.0678**
Baseline	MT-TF									
	Δ HR@1	Δ HR@2	Δ HR@3	Δ HR@4	Δ HR@5	Δ Recall@1	Δ Recall@2	Δ Recall@3	Δ Recall@4	Δ Recall@5
LSTM	**0.1100***	**0.1100***	**0.1100***	**0.0383***	**0.0191***	**0.0610***	**0.0768***	**0.0717***	0.0158	-0.0222
RLSTM	**0.1005***	**0.1100***	**0.1100***	**0.0383***	**0.0191***	**0.0578***	**0.0768***	**0.0717***	0.0158	-0.0222
TF	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	**0.0367***	**0.1632***	**0.2937***	**0.3330***
NQCFG	**0.0622***	0.0096	0.0000	0.0000	0.0000	**0.0331***	**0.0157***	-0.0089	0.0006	0.0106

Table 6. RD@N and DV@N improvements over the baselines on **MovieLens** dataset. The improvements that are statically significant are in bold. The value indicated with a star (*) is a positive and statistically significant value.

Baseline	MT-CNN									
	Δ RD@1	Δ RD@2	Δ RD@3	Δ RD@4	Δ RD@5	Δ DV@1	Δ DV@2	Δ DV@3	Δ DV@4	Δ DV@5
LSTM	**0.0321***	**0.0255***	**0.0191***	**0.0141***	**0.0112***	**-0.1445**	**-0.1168**	**-0.0881**	**-0.0701**	**-0.0600**
RLSTM	**0.0321***	**0.0255***	**0.0191***	**0.0141***	**0.0112***	**-0.1445**	**-0.1168**	**-0.0881**	**-0.0701**	**-0.0600**
TF	**0.0321***	**0.0253***	**0.0188***	**0.0138***	**0.0109***	**-0.1445**	**-0.1158**	**-0.0867**	**-0.0687**	**-0.0587**
NQCFG	-0.0036	-0.0012	-0.0016	-0.0005	-0.0002	0.0116	0.0043	0.0078	0.0054	0.0053

Baseline	MT-TF									
	Δ RD@1	Δ RD@2	Δ RD@3	Δ RD@4	Δ RD@5	Δ DV@1	Δ DV@2	Δ DV@3	Δ DV@4	Δ DV@5
LSTM	0.0009	**0.0014***	**0.0046***	**0.0055***	**0.0060***	-0.0036	**-0.0064**	**-0.0208**	**-0.0250**	**-0.0283**
RLSTM	0.0009	**0.0014***	**0.0046***	**0.0055***	**0.0060***	-0.0036	**-0.0064**	**-0.0208**	**-0.0250**	**-0.0283**
TF	0.0009	**0.0012***	**0.0043***	**0.0052***	**0.0057***	-0.0036	**-0.0053**	**-0.0194**	**-0.0236**	**-0.0269**
NQCFG	**-0.0348**	**-0.0253**	**-0.0160**	-0.0090	**-0.0054**	**0.1525***	**0.1147***	**0.0751***	**0.0505***	**0.0371***

Table 7. RD@N and DV@N improvements over the baselines on **Amazon** dataset. The improvements that are statically significant are in bold. The value indicated with a star (*) is a positive and statistically significant value.

Baseline	MT-CNN									
	Δ RD@1	Δ RD@2	Δ RD@3	Δ RD@4	Δ RD@5	Δ DV@1	Δ DV@2	Δ DV@3	Δ DV@4	Δ DV@5
LSTM	**0.0256***	**0.0296***	**0.0172***	0.0013	**0.0047***	**-0.1109**	**-0.1250**	**-0.0800**	-0.0056	**-0.0197**
RLSTM	**0.0256***	**0.0296***	**0.0172***	0.0013	**0.0047***	**-0.1109**	**-0.1250**	**-0.0800**	-0.0056	**-0.0197**
TF	**-0.0187**	**-0.0065**	-0.0015	**-0.0047**	**-0.0067**	**0.0714***	**0.0230***	0.0036	**0.0175***	**0.0261***
NQCFG	0.0031	**0.0096***	**0.0126***	**0.0070***	**0.0038***	**-0.0266**	**-0.0499**	**-0.0575**	**-0.0320**	**-0.0188**

Baseline	MT-TF									
	Δ RD@1	Δ RD@2	Δ RD@3	Δ RD@4	Δ RD@5	Δ DV@1	Δ DV@2	Δ DV@3	Δ DV@4	Δ DV@5
LSTM	**0.0387***	**0.0263***	**0.0092***	-0.0027	0.0025	**-0.1415**	**-0.0990**	**-0.0403**	**0.0147***	-0.0069
RLSTM	**0.0387***	**0.0263***	**0.0092***	-0.0027	0.0025	**-0.1415**	**-0.0990**	**-0.0403**	**0.0147***	-0.0069
TF	**-0.0056**	**-0.0098**	**-0.0095**	**-0.0087**	**-0.0090**	**0.0408***	**0.0490***	**0.0433***	**0.0379***	**0.0389***
NQCFG	**0.0162***	**0.0062***	**0.0046***	**0.0030***	0.0016	**-0.0572**	**-0.0238**	**-0.0177**	**-0.0117**	-0.0060