University of Reading

Department of Computer Science

Explainable Visually-Aware Recommender Systems Based on Heterogeneous Information Networks

Thanet Markchom

Submitted in Partial Fulfilment of the Requirement
of the Degree of Doctor of Philosophy

Declaration

I, Thanet Markchom, of the Department of Computer Science, University of Reading, confirm that this is my own work and figures, tables, equations, code snippets, artworks, and illustrations in this report are original and have not been taken from any other person's work, except where the works of others have been explicitly acknowledged, quoted, and referenced. I understand that failing to do so will be considered a case of plagiarism. Plagiarism is a form of academic misconduct and will be penalized accordingly.

I give consent to a copy of my report being shared with future students as an exemplar.

I give consent for my work to be made available more widely to members of UoR and the public with an interest in teaching, learning, and research.

Thanet Markchom

January 29, 2024

Abstract

Recommender systems are crucial for addressing information overload by providing users of online platforms, applications, or services with relevant item suggestions. Visually-aware systems enhance recommendations using images in various domains, e.g., fashion and entertainment. However, most visually-aware recommender systems lack explainability due to their reliance on black-box approaches such as deep-learning techniques. This thesis addresses the need for accurate and explainable visually-aware recommender systems by integrating visual information into Heterogeneous Information Networks (HINs). Three main research questions arise: How can the effective integration of visual information into HINs be achieved? How can the development of explainable visually-aware recommender systems based on HINs be approached? What methods can be used to enhance the explainability of HIN-based recommender systems?

To achieve these, this thesis proposes a three-part solution. Firstly, visually-augmented HINs are constructed by introducing visual factor nodes and relations, generated from various image features. A user representation learning method is introduced to combine semantic and visual preferences. Secondly, a Scalable and Explainable Visually-aware Recommender System (SEV-RS) framework is presented. SEV-RS utilizes meta-paths for explainability, enhancing recommendations using scalable feature extraction from visually-augmented HINs. Lastly, a meta-path translation model is introduced to improve the explainability of meta-path based systems, enhancing the performance of the proposed framework.

Extensive experiments with real-world datasets in movies and clothing domains assessed the effectiveness of visually-augmented HINs. These augmented HINs, along with the proposed user representation learning method, were employed in a recommendation model using Collaborative Filtering with K-Nearest Neighbors (CF-KNN). This model was compared with other CF-KNN models based on regular HINs. The findings indicate the efficacy of visually-augmented HINs and the representation learning approach in enhancing CF-KNN. They also exhibit the practicality of using visually-augmented HINs in state-of-the-art recommender systems. The evaluation of SEV-RS involved comparisons with state-of-the-art models using real-world and synthetic datasets. The results reveal that SEV-RS yields recommendations with high accuracy and explainability while requiring notably less computational time compared to other deep-learning models. The proposed meta-path translation approach was evaluated on two newly generated datasets derived from real-world recommendation data. It was compared with various sequence-to-sequence models in the task of translating a given meta-path to a group of meta-paths with higher explainability. The experiments validate its capability to generate more comprehensible alternative explanations for complex meta-paths. These approaches create pathways for developing recommender systems that address users' visual preferences and offer explanations based on understandable meta-paths. Implementing these methods could enhance user experiences, leading to more engaging and personalized recommender systems, with potential adaptability for improving other graph-utilizing explainable artificial intelligence applications.

Keywords: Recommender System, Visually-Aware Recommender System, Explainability, Heterogeneous Information Network, Meta-Path

Acknowledgements

I would like to sincerely acknowledge and convey my deepest gratitude to my supervisor, Dr. Huizhi Liang, for her unwavering guidance, support, and invaluable feedback throughout my PhD journey. Her expertise, encouragement, and dedication have been essential to the completion of this research.

I am extremely grateful to Professor James Ferryman, my co-supervisor, for consistently monitoring my progress, offering valuable advice, and ensuring I stayed on the right path, enabling me to complete this thesis.

I am also thankful to my assessors, Professor Xia Hong and Dr. Hong Wei, for their guidance and oversight during every PhD monitoring meeting.

My appreciation also goes to Dr. Varun Ojha for his valuable suggestions on the SemEval paper and this thesis, and to Dr. Jiaoyan Chen for his valuable suggestions that improved the SemEval paper.

Many thanks to the post-doctoral researchers and fellow graduate students who collaborated with me and provided support. While I may not be able to acknowledge each one individually, please know that your contributions and insightful discussions elevated both the quality of my research and my knowledge.

Special thanks to the anonymous reviewers for valuable comments that improved the quality of the manuscripts submitted during the course of my PhD study and this thesis.

I would also like to acknowledge the Department of Computer Science at the University of Reading for providing the essential resources and environment for effective research.

Deep gratitude goes to the Development and Promotion of Science and Technology Talents Project (DPST), a Royal Government of Thailand scholarship, for their financial support.

Finally, my heartfelt appreciation goes to my family and friends for their unwavering support and motivation. Their encouragement, understanding, and unwavering love have been the driving force behind my achievements.

Contents

List of Figures

List of Tables

Glossary

BPR	Bayesian Personalized Ranking
BPR-MF	Bayesian Personalized Ranking Matrix Factorization
BRIEF	Binary Robust Independent Elementary Features
Caffe	Convolutional Architecture for Fast Feature Embedding
CF-KNN	Collaborative filtering recommendation method using K-Nearest Neighbors
CNN	Convolutional Neural Network
FAST	Features from Accelerated Segment Test
GDPR	General Data Protection Regulation
HIN	Heterogeneous Information Network
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
KGAT	Knowledge Graph Attention Network
KNN	K-Nearest Neighbors
LSTM	Long short-term memory network
MF	Matrix Factorization
ORB	Oriented FAST and Rotated BRIEF
QCFG	Quasi-Synchronous Context-Free Grammar
Seq2Seq	Sequence-to-Sequence
SEV-RS	Scalable and Explainable Visually-aware Recommender System
SIFT	Scale-Invariant Feature Transform
SURF	Speeded Up Robust Features
VBPR	Visual Bayesian Personalized Ranking model

List of Symbols

$\mathrm{A_p}(m)$	Mean Average Precision@N value of the recommendations based on a meta-path
$\mathrm{A_p}(m^*)$	Mean Average Precision@N value of the recommendations based on an explainable meta-path
$\mathrm{A_r}(m)$	Recall@N value of the recommendations based on a meta-path
$\mathrm{A_r}(m^*)$	Recall@N values of the recommendations based on an explainable meta-path
A	Non-terminal node in a target tree in a meta-path quasi-synchronous context-free grammar
$a_{p,m}$	Item-MetaPath association
$a_{u,m}$	User-MetaPath association
B	Non-terminal or pre-terminal node in a target tree in a meta-path quasi-synchronous context-free grammar
b_w	Terminal bias
$\mathrm{C}(m)$	Meta-path credibility of a meta-path
C	Non-terminal or pre-terminal node in a target tree in a meta-path quasi-synchronous context-free grammar
$C(k, k+1)$ ...	Probability of nodes with the $(k+1)$th node type in a meta-path given nodes with the kth node type (i.e., the previous node type) in a meta-path
$\mathbf{c}_{\alpha_i k}$	Memory cell vector of the kth child in a target tree
$\mathcal{D}_S$	Training sample set
$\mathrm{D}(m)$	Meta-path diversity
D	Pre-terminal node in a target tree in a meta-path quasi-synchronous context-free grammar
d_i	Haar wavelet response in the horizontal direction
d_j	Haar wavelet response in the vertical direction
$\mathbf{E}_P$	Latent space projection matrix of an item meta-path feature
$\mathbf{E}_U$	Latent space projection matrix of a user meta-path feature
$\mathbb{E}$	Set of terminals which is the set of node types in a meta-path quasi-synchronous context-free grammar

E_{up}	Explainability score between a user and an item
$\mathbf{e}_{\mathsf{A}[\alpha_i]}$	Embedding vector of a node in the target tree of a target meta-path that is transduced by the node in the source tree of a source meta-path
e_n, $e_{n'}$, e_{n_i}	Embedding vector of a node
$\mathbf{f}$	Image feature vector
$\mathbf{f}_p$	Item meta-path feature
$\mathbf{f}_u$	User meta-path feature
f	Function that maps a meta-path to a set comparable explainable meta-paths
f_1, f_2, f_3, f_4	Feedforward neural networks with residual layers
$\mathbb{G}$	HIN schema
$\mathbb{G}'$	Visually-augmented HIN schema
$\mathcal{G}$	HIN
$\mathcal{G}'$	Visually-augmented HIN
$\mathbb{G}[t]$	Meta-path quasi-synchronous context-free grammar
$g(m)$	Global connectivity of a meta-path
$\mathbf{H}$	Second-order Hessian matrix
$\mathbf{h}_{\alpha_i k}$	Hidden state vector of the kth child in a target tree
$\mathbf{h}_{\alpha_i}$	Token embedding of a node in the target tree of a target meta-path
$h(\mathbf{f}_u, \mathbf{f}_p)$	Cosine similarity between a user meta-path feature and an item meta-path feature
$\mathbf{I}$	Image
K_m	Negative sample size in metapath2vec
K_t	Number of target trees sampled in the meta-path translation model
k^*	Number of representative visual factors per user
k_v	Number of visual factors
$L(\theta, \phi)$	Loss function of meta-path translation model
l	Meta-path length
$\mathcal{M}$	Set of meta-paths
$\mathcal{M}^*$	Set of explainable meta-paths
m	Meta-path
m'	Probabilistic meta-path
m_i	The ith meta-path in a set of meta-paths
$\mathbb{N}$	Node types set
$\mathbb{N}'$	Node types set in a HIN schema
$\mathcal{N}$	Nodes set
$\mathcal{N}'$	Nodes set in a visually-augmented HIN

$\mathcal{N}_i, \mathcal{N}_j$	Set of nodes with the ith/jth node type in a meta-path
$\mathcal{N}_i^n$	Set of neighbor nodes of a given node with the ith node type in a meta-path
$\mathcal{N}_u^p$	Set of a given user's neighbors who have interacted with a given item
$\mathbb{N}$	Set of non-terminals that expand to other non-terminals in meta-path quasi-synchronous context-free grammar
N_i	The jth node type
N_j	The ith node type
n	Node
n_i	Node with the ith node type in a meta-path
n_j	Node with the jth node type in a meta-path
$\mathcal{P}$	Items set
$\mathcal{P}_u$	Set of items interacted by a given user
$\mathcal{P}_u^+$	Set of positive items of a given user
$\mathbb{P}$	Set of non-terminals that expand to terminals (i.e. pre-terminals) in a meta-path quasi-synchronous context-free grammar
P	Power set
$Pr(z)$	Probability of a given path instance
$\mathbf{p}$	Final item latent factors
$p_\theta(r)$	Probability of a rule in a meta-path quasi-synchronous context-free grammar
$p_\phi(t\|m)$	Distribution function of the source tree given a source meta-path
$p_\theta(t^*\|t)$	Probability of the target tree given a source tree
$\mathbb{R}$	Relation types set
$\mathbb{R}'$	Relation types set in a visually-augmented HIN schema
$\mathcal{R}$	Relations set
$\mathcal{R}'$	Relations set in a visually-augmented HIN
$\mathbb{R}[t]$	Set of context-free rules conditioned on a source tree in a meta-path quasi-synchronous context-free grammar
$\mathfrak{R}$	Set of real numbers
$\mathrm{R}(m)$	Meta-path readability of a given meta-path
$R_{N_iN_j}$	Relation type from the ith node type to the jth node type in a meta-path
R_{N_i,N_j}^{-1}	Inverse relation type from the jth node type to the ith node type in a meta-path
R_{PV}	Item-visual factor relation type
R_{UV}	User-visual factor relation type
$\mathcal{R}_V$	Visual relations set
$r_{x,y}$	Relation from one node to another

r	Context-free rule conditioned on a source tree in a meta-path quasi-synchronous context-free grammar
$\mathbf{S}$	Integral image
$\mathbb{S}$	Distinguished start symbol in a meta-path quasi-synchronous context-free grammar
$s(u, p, m)$	Meta-path based connectivity strength
$sim(u, u_k)$	Similarity between one user and another
$\mathcal{T}(m)$	Set of trees whose yields are a given meta-path
$\mathcal{T}(m^*)$	Set of trees whose yields are a given explainable meta-path
t	Parse tree of a meta-path (a source tree)
$\hat{t}$	Maximum a posteriori probability tree obtained from the arguments of the maxima of the conditional probability function of a source tree given a source meta-path
$\tilde{t}$	Sample from the distribution of a source tree given a source meta-path
t^*	Parse tree of an explainable meta-path (a target tree)
t_f	FAST keypoint validation threshold
$\mathcal{U}$	Users set
$\mathcal{U}_p$	Set of users interacted with a given item
$\mathbf{u}$	Final user latent factors
$\mathbf{u}_\mathsf{A}$	Embedding of a node in the source tree of a source meta-path
$\mathbf{u}_\mathbb{S}$	Embedding of the distinguished start symbol in the meta-path grammar
$\mathbf{u}_w$	Embedding of a node that is a terminal in the source tree of a source meta-path
$\mathcal{V}$	Visual factor nodes set
V	Visual factor node type
$\mathbf{v}_i$	The ith visual factor
$\mathbf{v}^u$	User visual preference profile
v_i	The ith visual factor node that corresponds with the ith visual factor
$\mathbb{W}$	Weight function of relation types in a relation types set in a HIN schema
$\mathbb{W}'$	Weight function for relation types in a relation types set in a visually-augmented HIN schema
$w(x, y)$	Weight of a relation
w	Terminal node in a target tree in a meta-path quasi-synchronous context-free grammar
w_d	Weight decay hyperparameter for the proposed meta-path translation model
X	Distribution from which a negative node is drawn in metapath2vec

X_j	Distribution from which a negative node with the jth node type in a meta-path is drawn in metapath2vec
$\mathbf{x}_{\alpha_i}$	Token embedding of a node in the target tree of a target meta-path
$\mathbf{x}^*_{\alpha_i}$	Positional embedding of a node in the target tree of a target meta-path
$\mathbf{x}'_{\alpha_i}$	Latent feature embedding of a node in the target tree of a target meta-path
x	Node
$\hat{x}_{up}$	Recommendation score of a given user towards a given item
y	Node
$\mathcal{Z}_m$	Set of path instances of a meta-path
z	Path instance
α	Global offset
α_i	Node in a source tree
$\boldsymbol{\beta}_P$	Item feature bias
$\boldsymbol{\beta}_U$	User feature bias
β_p	Item bias term
β_u	User bias term
$\boldsymbol{\gamma_p}$	Item traditional latent factors
$\boldsymbol{\gamma_u}$	User traditional latent factors
δ	Probability of probabilistic meta-path
δ_1	Pre-defined precision threshold
δ_2	Pre-defined recall threshold
δ_C	Credibility threshold
δ_D	Diversity threshold
δ_R	Readability threshold
κ	Maximum length for selecting explainable meta-paths
Θ	BPR-MF model parameters
Θ_m	Metapath2vec model parameters
θ	Parameters of the rule probabilities in a meta-path quasi-synchronous context-free grammar
$\boldsymbol{\theta_p}$	Item meta-path based latent factors
$\boldsymbol{\theta_u}$	User meta-path based latent factors
λ	Explainability regularization hyperparameter of the proposed meta-path translation model
λ_E	Explainability regularization hyperparameter of the proposed scalable and explainable recommender system
λ_Θ	Regularization hyperparameter

σ	Sigmoid function
ς	Softmax function
ϕ	Node type mapping function
ψ	Relation type mapping function
ζ	Threshold for selecting keypoints in SIFT

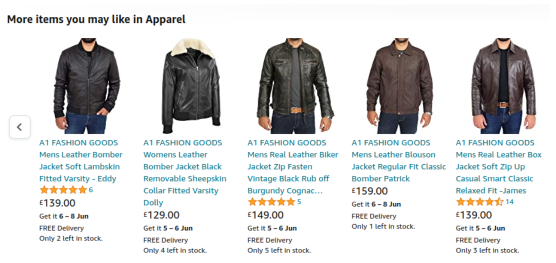
Chapter 1

Introduction

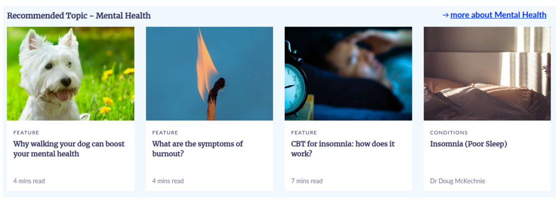
1.1 Background

The massive increase in available online information has resulted in an abundance of options for users. As a result, users have been facing challenges in identifying certain pieces of information that are both personalized and pertinent to their preferences. Recommender systems have emerged as an essential tool in resolving this problem by suggesting information that potentially matches users' interests [6]. They have been extensively employed in several industries and online platforms such as e-commerce, e-health, and e-learning to enhance the user experience by facilitating content discovery [7]. In e-commerce, for example, recommender systems can help users discover new products that they may be interested in. In the field of e-health, recommender systems have the potential to elevate the quality of care and streamline healthcare delivery by offering patients personalized information and services. In e-learning, recommender systems can suggest relevant courses or resources to learners, improving their learning outcomes.

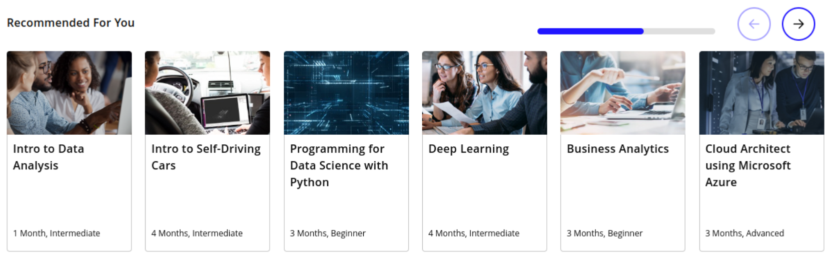
Figure 1.1 illustrates examples of recommender systems implemented on three platforms across different domains including e-commerce, e-health, and e-learning. Figure 1.1a depicts a case from the renowned Amazon website [1], recognized as one of the world's largest online retailers. In this figure, Amazon's recommendation engine suggests products based on users' browsing and purchase history. It recommends jackets similar to the ones previously interacted with by the user. Figure 1.1b shows a snapshot from the healthcare website called Patient [2], offering diverse content and resources to aid users in managing their health. In this figure, the Patient's recommendation feature offers a health-related topic that potentially matches the user's interests and needs and recommends relevant articles that are popular in that topic.


More items you may like in Apparel
A1 FASHION GOODS Mens Leather Bomber Jacket Soft Lambskin Fitted Varsity - Eddy
6
£139.00
Get it 6 – 8 Jun
FREE Delivery
Only 2 left in stock.
A1 FASHION GOODS Womens Leather Bomber Jacket Black Removable Sheepskin Collar Fitted Varsity Dolly
£129.00
Get it 5 – 6 Jun
FREE Delivery
Only 4 left in stock.
A1 FASHION GOODS Mens Real Leather Biker Jacket Zip Fasten Vintage Black Rub off Burgundy Cognac…
5
£149.00
Get it 5 – 6 Jun
FREE Delivery
Only 5 left in stock.
A1 FASHION GOODS Mens Leather Blouson Jacket Regular Fit Classic Bomber Patrick
£159.00
Get it 6 – 8 Jun
FREE Delivery
Only 1 left in stock.
A1 FASHION GOODS Mens Real Leather Box Jacket Soft Zip Up Casual Smart Classic Relaxed Fit -James
14
£139.00
Get it 5 – 6 Jun
FREE Delivery
Only 3 left in stock.

(a) e-commerce


Recommended Topic – Mental Health
→ more about Mental Health
FEATURE
Why walking your dog can boost your mental health
4 mins read
FEATURE
What are the symptoms of burnout?
4 mins read
FEATURE
CBT for insomnia: how does it work?
7 mins read
CONDITIONS
Insomnia (Poor Sleep)
Dr Doug McKechnie

(b) e-health



(c) e-learning

Figure 1.1: Examples of recommender systems in (a) e-commerce [1], (b) e-health [2], and (c) e-learning websites [3]. These examples illustrate lists of recommendations showing to a user on their websites.

Lastly, Figure 1.1c showcases a screenshot from the Udacity website [3], a well-known e-learning platform that provides a variety of courses and programs in various subjects. Similarly, this figure presents Udacity's recommendation capabilities, which assist users in discovering suitable courses and programs aligned with their learning preferences.

Recommender systems are also valuable for businesses. By understanding the preferences and behavior of their customers, businesses can tailor their offerings to meet their customers' needs, leading to increased loyalty and customer retention [8]. Furthermore, recommender systems can enhance the user experience by reducing the time and effort required to find relevant information [6]. Overall, recommender systems offer a range of advantages that span various domains. By tailoring recommendations to individual preferences, they create personalized experiences, promote engagement and exploration of content or products, and narrow down choices, thus saving users' time and guiding their decisions. These advantages highlight the significant role of recommender systems in online communities. Therefore, ongoing advancements are necessary to ensure user satisfaction and maintain relevance in diverse online contexts [9, 10, 11].

Many existing recommender systems depend largely on user-item interactions to gain insights into users' preferences [12, 13]. For instance, a collaborative filtering model based on

K-nearest neighbors makes accurate recommendations by leveraging similarities between users and uncovering intricate preferences hidden among like-minded users [14]. This approach has been widely applied in various applications due to its simplicity in terms of implementation and serves as a foundation for more advanced recommender systems [12]. However, this approach can prove to be highly inadequate in scenarios where the interactions are sparse or not available at all [12, 13]. For instance, when a new user is added to the system, his/her previous interaction data may not be sufficient to generate accurate recommendations. This problem is commonly referred to as the cold-start problem [14] and can significantly reduce the effectiveness of recommender systems. To address such an issue, additional information such as user and item metadata has been incorporated into the recommendation process to compensate for the lack of user historical data [13]. By leveraging this auxiliary information, the connectivity between users and items can be enriched, and the accuracy of recommendations can be significantly improved. This approach has been shown to be highly effective in overcoming the sparsity of interaction data and cold-start problems. As a result, the use of side information has become an essential technique for enhancing the performance of recommender systems in diverse applications [13, 15, 16].

Besides semantic information derived from user and item metadata, visual information from images also provides useful knowledge inferring users' individual preferences [16]. Since an image can provide numerous information compared to a single word, it is intuitively capable of providing rich information about users' preferences [17]. Visual information can be retrieved from an image in the form of visual features, encompassing distinct characteristics that define an image [18]. These features can be broadly categorized into various types including color features, texture features, and shape features [19]. Color features involve the identification and quantification of different hues, e.g., color co-occurrence matrix, color histogram, and color moments [20]. Texture features describe patterns and variations in pixel intensities. Examples of texture features include features extracted from Gabor filtering [21] and wavelet transform [22]. Shape features refer to quantitative measures or descriptors that characterize the geometric properties of an object's outline or boundary within an image. These include roundness, eccentricity, centroid, minimum bounding rectangle, moments, orientation, etc [23]. Apart from these basic features, more advanced feature extraction algorithms have been developed to extract more comprehensive features [24] such as SIFT (Scale-Invariant Feature Transform) [25], SURF (Speeded-Up Robust Features) [26], and ORB (Oriented FAST and Rotated BRIEF) [27]. These algorithms specialize in capturing detailed information from specific regions, known as key points, within an image. The key points identified by these algorithms not only describe the local textures and shapes but also contain spatial coordinates, indicating the position of distinctive features within the image.

The advancement of deep learning has further significantly improved the extraction of visual features. Deep learning techniques such as convolutional neural networks (CNNs) [28] and recurrent neural networks (RNNs) [29] excel at learning more complex and abstract visual features by capturing patterns and structures within an image. CNNs, with their hierarchical layers, automatically extract and combine different features, enabling the network to learn intricate patterns and representations in an image [30]. On the other hand, RNNs are well-suited for capturing temporal dependencies and dynamic aspects of visual features [31]. However, deep learning models that learn such complex visual features may operate as black-box systems, posing a challenge in interpreting their decision-making processes and what these extracted features exactly represent [32]. These learned features are typically described as latent visual features, meaning that they are not readily apparent or observable in the raw input images but captured through complex relationships, structures, or patterns [33, 34]. For instance, a neural network trained on images of faces might extract latent visual features representing facial characteristics, even if these features are not obvious or explicitly defined [35]. These latent visual features play a pivotal role in enhancing the performance of computer vision models [34]. By capturing nuanced and abstract information, they contribute to the model's capacity to generalize effectively and make accurate predictions across a diverse range of data [34]. However, the inherent lack of interpretability in these latent visual features can be a drawback, particularly in applications where transparency and explainability are important [32]. Striking a balance between the powerful predictive capabilities facilitated by latent visual features and the need for interpretability remains a crucial consideration in the usage of visual features [32, 36].

Similarly to computer vision models, numerous modern recommender systems utilize visual features to improve their capability [37, 38, 39, 40, 41, 42, 43]. Recommender systems that leverage such information can be referred to as *visually-aware recommender systems* [38, 44, 45, 46]. These systems are designed to provide more accurate and personalized recommendations by considering not only users' previous interactions with items but also the visual information of these items. Visually-aware recommender systems are particularly useful in scenarios where users' preferences are strongly influenced by the visual characteristics of the items. For example, in the context of fashion e-commerce, users may be interested in items

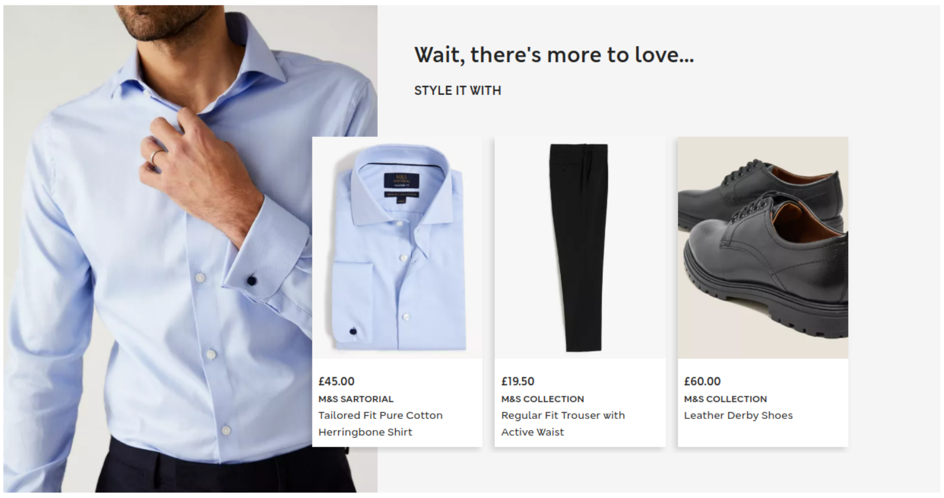

Wait, there's more to love...
STYLE IT WITH
£45.00
M&S SARTORIAL
Tailored Fit Pure Cotton Herringbone Shirt
£19.50
M&S COLLECTION
Regular Fit Trouser with Active Waist
£60.00
M&S COLLECTION
Leather Derby Shoes

Figure 1.2: Example of visually-aware recommendations on a clothing retailer website [4] which involves a situation where a user expresses interest in purchasing a blue shirt depicted in an image. The recommender system then suggests two items to the user—black trousers and black leather shoes—that appear compatible and visually complement the shirt.

with a particular style, color, or pattern. Figure 1.2 shows an example of a visually-aware recommendation on an e-commerce fashion website. In this example, the user is interested in a blue shirt. The recommender system then recommends two items which are black trousers and black leather shoes since their styles are compatible/complementary to the blue shirt. One of the key challenges in developing visually-aware recommender systems is how to effectively extract visual features from items and integrate them into the recommendation process. Most visually-aware recommender systems adopt pre-trained computer vision models to extract latent visual features from item images [37, 38, 40]. Then, they integrate them into their recommendation learning process as additional information along with user-item interactions to learn users' preferences more effectively. For example, VBPR [37] uses latent visual features extracted from a pre-trained CNN model to effectively factorize users and items for learning recommendations. DVBPR [38] enhances the performance of VBPR by simultaneously training a recommendation model and a CNN model, facilitating the extraction of more effective visual latent features. aVBPR [41] combines the original VBPR model, which models users' visual preferences, with models for users' sequential behaviors and non-visual features. This extension enhances the capability of the original VBPR model. These previous studies demonstrate the successful utilization of visual features to enhance recommendation accuracy, surpassing reliance solely on user-item interactions. However, since they are latent features and are commonly high-dimensional, it can be difficult to understand what these visual features really represent [40, 45]. This lack of clarity in understanding what these features truly signify contributes to the low explainability of visually-aware recommender systems. Consequently, users may encounter difficulty in comprehending the rationale behind the recommendations generated by these systems [40, 45].

Recent studies have highlighted potential issues that can arise from using artificial intelligence (AI) models, including recommender systems, without proper comprehension [47]. These issues include biases, ethical violations, and other serious consequences [48]. To address these concerns, regulations such as the General Data Protection Regulation (GDPR) [49] have been established in various countries. These regulations are employed to ensure fairness and users' trust in AI systems. In response to the explainability regulations, research in explainable AI has emerged and has been continuously receiving a large amount of attention. In general, explainable AI can provide explanations or allow humans to understand the logic behind its decision-making process. During the past decade, there have been efforts to develop interpretable machine learning models in which their internal mechanisms are comprehensible for humans. These models include linear/logistic regression models, decision trees, and Bayesian networks. Apart from these interpretable models, ad-hoc explanation generation methods have been proposed to explain predictions from black-box models. Some methods aim at generating a local explanation that explains the decision reason of a specific input such as LIME [50], Anchors [51], and SHAP [52]. Some ad-hoc methods aim to generate a global explanation that explains the overall logic of the model such as the Global Sensitivity Analysis method [53] and ASTRID [54].

Compared to machine learning models, it is more challenging to explain deep learning models due to their more complex architectures. In [55], Concept Activation Vectors (CAVs) were introduced to globally explain the internal mechanism of a neural network with human-friendly concepts. Automatic Concept-based Explanations (ACE) [56] was proposed to explain a neural network without manually defining concepts. Neural Additive Models (NAMs) [57] were proposed to train multiple neural networks where each neural network focuses on a single input feature. Some explaining methods have been developed specifically for CNNs such as a gradient-based method for generating saliency maps [58] and DeConvNet [59]. Despite the numerous prior efforts, explaining black box models is still challenging in many aspects. One of these aspects is that explaining methods used in a classification task may not be suitable for a recommendation task since a recommendation task is a user-personalized task. It could be more complex to explain than a classification task [60].

In the past, recommender systems were developed solely with the aim of improving their accuracy, without much consideration for how they worked. These systems were typically designed as black-box models, which means that their decision-making mechanisms were not transparent or understandable to either the developers or end-users [48]. Furthermore, with the advancement of deep learning, many modern recommender systems have been utilizing deep learning models to effectively learn recommendations. Despite the evident superiority in performance, the complexity of deep learning model architectures has led to significant challenges regarding their interpretability. These challenges ultimately obstruct the ability to understand the underlying rationale behind the recommendations. In response to these issues, researchers have turned their attention to developing *explainable recommender systems* that are transparent and can be understood. These recommender systems aim to provide explanations for their recommendations or encourage explainable recommendations to be predicted rather than non-explainable ones. They allow users to better understand the decision-making process, leading to increased trust and confidence in the systems. Another benefit of explainable recommender systems is that they can help mitigate biases and ethical concerns that may arise from the use of black-box models. By providing clear explanations for their recommendations, any biases that may be present in the data or algorithms can be identified and addressed accordingly. This ensures that the recommendations are fair, unbiased, and ethically sound. Moreover, explainable recommender systems also provide greater control and flexibility to users, as they can adjust their preferences and settings based on the explanations provided. This can help to improve the accuracy and relevance of the recommendations, as users can provide feedback and adjust their preferences as needed. Due to these advantages, the development of explainable recommender systems has become a major area of research in recent years [10] and is expected to play an increasingly important role in ensuring that AI models are used ethically and responsibly in the future.

Several resources have been utilized to achieve explainability in recommendations. One of the ubiquitous resources is *heterogeneous information networks* [5, 13, 15, 61, 62]. A heterogeneous information network (HIN) [63] is a specialized type of graph structure designed to model complex relationships between entities. In contrast to traditional homogeneous networks, which primarily involve nodes of the same type and a single type of relationship, HINs accommodate diverse entities with varying types and multiple types of relationships between them. In HINs, nodes serve as representations of entities, and these entities can exhibit diversity in terms of type. For example, in a recommendation scenario, nodes could represent users, items, genres, or any other relevant entity [5, 61, 62]. Edges (or relations) represent relationships between these entities and can denote different types of connections, such as user-item interactions, user-demographic associations, or item-metadata relationships [5, 61, 62]. One key strength of HINs lies in their ability to capture the rich and intricate relationships present in real-world systems [13]. By allowing for nodes of different types and diverse relationships, HINs offer a more nuanced and accurate representation of the underlying complexity in various domains [15]. This makes them particularly valuable in scenarios where entities have multiple facets and interact through diverse relationships.

HINs play a pivotal role in advancing the explainability of recommender systems by incorporating diverse sources of information [11, 62, 64, 65, 66]. Unlike traditional recommender systems that predominantly rely on user-item interactions, HIN-based recommender systems leverage a variety of interconnected entities and relationships via *multi-hop relations* [5, 65, 67, 68]. A multi-hop relation represents a high-order connection between two nodes that are not adjacent. They have been extensively used to facilitate recommendation generation and improve the explainability of recommendations [65, 67, 68, 69]. For instance, Figure 1.3 shows an example of a HIN consisting of four node types, i.e., user, item, category, and brand node types, and three relation types, i.e., user-item, item-category, and item-brand relation types (all relation types are bidirectional). There is a multi-hop relation between "User 1" and "T-shirt D" through a path "User1" - "T-shirt C" - "Category: T-shirt" - "T-shirt D". Based on this path, it is possible to recommend "T-shirt D" to "User 1" since they are linked through a high-order connection. Also, the meta-path that connects them can serve as an explanation. This explanation can be interpreted as "T-shirt D" is recommended to "User 1" because it is in the same category as one of "User 1"'s previously bought items. As in this example, multi-hop relations provide insights that other types of resources (e.g., texts or images) cannot provide. Although texts and images provide valuable information, multi-hop relations in HINs offer a distinct advantage due to their ability to unravel complex connections within the networks [70]. Specifically, in contrast to images, which typically provide only isolated visual information, HINs offer interrelatedness information of diverse entities [13]. Due to this unique advantage, HINs have therefore been widely leveraged to develop modern explainable recommender systems during the past decade.

To effectively utilize multi-hop relations, meta-paths have been proposed to extract semantically meaningful multi-hop relations in a HIN under certain assumptions [13]. A meta-

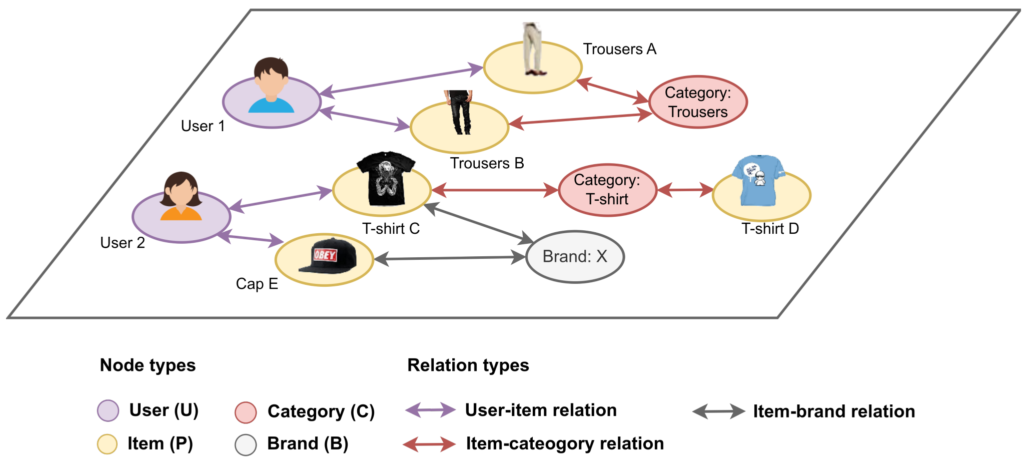


Figure 1.3: Example of a HIN in a clothing recommendation scenario consisting of four node types, i.e., user, item, category, and brand node types, and three relation types, i.e., user-item, item-category, and item-brand relation types (all relation types are bidirectional)

path [71] is defined as a sequence of node types and relation types. It is used to discover semantic node connectivity based on a sequence of multi-typed factors in a HIN. When extracting multi-hop relations based on a meta-path, it is essential to traverse a path in accordance with the defined meta-path to establish a connection between two distinct entities. This approach facilitates the capture of specific and intricate high-order interactions that span multiple node and relation types. For example, given a HIN in Figure 1.3, let U, P, C and B denote user, item, category, and brand node types respectively. A meta-path $UPCP$ represents a multi-hop relation between U and P. It involves a sequence of relationships: from users to items, then from items to categories, and finally from categories back to items. As can be seen in this example, by using meta-paths, multi-hop relations under specific assumptions are extracted while disregarding other multi-hop relations with different semantic meanings. This approach enables the incorporation of more specific high-order information in the process of learning users' preferences. As such information employed for learning users' preferences is explicit, it becomes possible to generate recommendations and provide explanations based on the meanings associated with the meta-paths utilized [72, 73]. Due to this unique advantage, HIN-based recommender systems using meta-paths have become ubiquitous.

To summarize, several existing visually-aware recommender systems have excelled at leveraging visual features extracted from images to enhance recommendation accuracy [16]. Nevertheless, current research on recommender systems has been focusing on not only accuracy but also the explainability aspect of the systems [10]. This additional aspect has been considered to increase users' trust and confidence, ensure fairness and ethics, and provide greater control and flexibility in recommender systems [10, 47]. Consequently, enhancing explainability in visually-aware recommender systems is undoubtedly necessary and requires further exploration [16]. As HINs can offer explainability in recommendations, using them in visually-aware recommender systems can intuitively lead to better explainability of their recommendations. Despite the achievements of current HIN-based recommender systems, particularly those utilizing meta-paths, they predominantly focus on leveraging semantic information such as user and item metadata within HINs. They typically overlook the potential of visual information [74]. This omission results in a significant gap in integrating visual information within HINs to enhance the recommendation learning process. By bridging this gap, HINs can play a pivotal role in facilitating visually-aware recommendations. Simultaneously, they can introduce explainability to these recommendations by leveraging complex network structures and relationships. This approach provides users with a deeper understanding of the reasoning behind the recommendations, thereby enhancing their confidence in the systems and aiding better-informed decisions.

1.2 Research Questions

Most visually-aware recommender systems utilize extracted latent image features in a black-box style. As a result, they typically suffer from explainability issues. Since explainability has become another focal point in developing modern recommender systems, how to develop visually-aware recommender systems that can achieve both high accuracy and high explainability is necessary. Inspired by the omnipresence of HIN-based explainable recommender systems, using HINs is one possible solution to facilitate the task of visually-aware recommendations with explainability. However, many existing HIN-based recommender systems typically ignore visual information. Integrating visual information into a HIN can be complicated and may not be as straightforward as semantic information such as user and item metadata [75]. The heterogeneity of image features compared to other data types, such as numerical or categorical data, can be an obstacle when combining and comparing image features with other types of features in a HIN. Furthermore, there may be semantic gaps between image features and other features in HINs. These gaps can cause difficulty in effectively modeling relationships between entities in a HIN. All of these challenges lead to the following research problems:

- How to effectively integrate visual information into a HIN so that visual information and multi-hop relations in the network can be simultaneously leveraged in visually-aware recommender systems?
- How to build recommender systems that can produce accurate and explainable visually-aware recommendations by using HINs?
- How to further improve the explainability of HIN-based recommender systems?

1.3 Objectives and Outcomes

In accordance with the research questions stated in Section 1.2, the objectives of this thesis are as follows:

- To develop a method that integrates visual features into a HIN so that both visual information and multi-hop relations can be leveraged simultaneously for learning recommendations and to validate this method using real-world data.
- To build an explainable visually-aware recommendation framework based on the HIN in which visual information is integrated and to evaluate the effectiveness of the proposed framework using real-world data.
- To develop a method that enhances the explainability of visually-aware recommender systems based on HINs and to validate its performance for potential real-world applications.

To achieve these objectives, this thesis delivers the following outcomes:

- Integration of visual information into HINs: A novel concept of visually-augmented HINs is proposed, integrating image features as visual factor nodes connected through visual relations. Various image feature extraction methods are explored, and these augmented networks are leveraged to enhance recommendation accuracy. Extensive experiments are conducted on real-world datasets, showing that visually-augmented HINs can improve recommendation accuracy.
- Development of explainable visually-aware recommender systems using HINs: A novel framework is proposed, using multi-hop relations in visually-augmented HINs for scalable and explainable recommendations. This framework integrates meta-path based features

and explainability for improved recommendation quality. The proposed framework is evaluated and proves its ability to produce accurate, explainable recommendations while maintaining scalability.

- Enhancing explainability of HIN-based recommender systems: A method to enhance the explainability of HIN-based recommender systems, utilizing meta-path based approaches, is introduced. This method can offer alternative explanations that are more comprehensible for meta-path based recommendations.
- Practical implications: The outcomes of this thesis contribute to the development of more user-engaging and personalized experiences in various domains including e-commerce and entertainment. The transparency and enhanced explanations provided by these methods can enhance user trust and comprehension of recommendations. Moreover, the proposed approaches offer the potential for adaptation and generalization to enhance other explainable AI applications, especially those that utilize graphs.

1.4 Thesis Outline

The rest of this thesis is structured as follows:

- **Chapter 2** provides a comprehensive literature review that covers traditional recommender systems, visually-aware recommender systems, explainable recommender systems, and HIN-based explainable recommender systems. The review explores state-of-the-art approaches in each of these topics, highlighting their strengths and limitations, and identifying gaps in the existing research. This part of the thesis has been submitted and is currently under review for potential publication as a review paper.
- **Chapter 3** focuses on the first research question and objective. It discusses the construction of visually-augmented HINs, where visual information is integrated into the network to facilitate the task of visually-aware recommendations. Furthermore, a visually-aware recommendation framework that leverages visually-augmented HINs is presented. This chapter describes the experiments conducted to evaluate this framework, including the methodology, results, and subsequent discussions. The results gained from the experiments contribute to the understanding of the effectiveness of visually-augmented HINs and the proposed visually-aware recommendation approach based on visually-augmented HINs.

- **Chapter 4** centers around the second research question and objective of this thesis. It introduces the proposed explainable visually-aware recommendation framework based on visually-augmented HINs discussed in the previous chapter. This chapter begins by presenting the proposed framework and proceeds to explain its components in detail. First, a meta-path based feature extraction method used in this framework is described. After that, this chapter delves further into the concept of meta-path based explainability and explains how the explainability scores of user-item pairs are computed. Based on the meta-path features and meta-path explainability, this chapter then describes the proposed approach for generating explainable recommendations. Finally, the experiments conducted to evaluate the proposed framework are discussed, including a comparison between the proposed framework and state-of-the-art recommendation models in terms of accuracy, explainability, and scalability. All of these provide valuable insights into the performance and effectiveness of the proposed framework in generating accurate and explainable recommendations.

- **Chapter 5** is dedicated to addressing the third research question and objective of this thesis. It focuses on enhancing the explainability of explainable visually-aware recommender systems using HINs. This chapter begins by introducing the concept of the meta-path translation task, which aims to improve the explainability of HIN-based recommendations, specifically meta-path based recommendations. A meta-path grammar is proposed to facilitate the process of translating meta-paths. Subsequently, this chapter describes the meta-path translation model proposed to accomplish this task. Furthermore, the construction of a meta-path translation dataset is explained, followed by a detailed account of the experiments conducted on two novel meta-path translation datasets generated in this thesis. Finally, this chapter presents and discusses the results obtained from comparing the proposed meta-path translation model with other state-of-the-art models. These results offer insights into the effectiveness and performance of the proposed approach for enhancing the explainability of visually-aware recommender systems based on HINs.

- **Chapter 6** serves as the conclusion of this thesis, where all the contributions of the work are summarized. Additionally, the limitations of the research are addressed, and potential areas for future work are discussed.

1.5 List of Publications

Certain chapters of this thesis have been published in the following peer-reviewed publications, with one manuscript currently under review:

- Chapter 3
  - T. Markchom and H. Liang, "Augmenting visual information in knowledge graphs for recommendations," in 26th International Conference on Intelligent User Interfaces, p. 475–479, 2021. https://doi.org/10.1145/3397481.3450686.
  - H. Liang and T. Markchom, "TNE: A general time-aware network representation learning framework for temporal applications," Knowledge-Based Systems, vol. 240, 2022. https://doi.org/10.1016/j.knosys.2021.108050.
- Chapter 4
  - T. Markchom, H. Liang, and J. Ferryman, "Scalable and explainable visually-aware recommender systems," Knowledge-Based Systems, vol. 263, p. 110258, 2023. https://doi.org/10.1016/j.knosys.2023.110258.
- Chapter 5
  - T. Markchom, H. Liang, and J. Ferryman, "Explainable Meta-Path Based Recommender Systems," ACM Transactions on Recommender Systems, 2023. https://doi.org/10.1145/3625828.

Chapter 2

Literature Review

This chapter presents a comprehensive literature review that includes a wide range of key topics in the field of recommender systems, including general recommender systems, visually-aware recommender systems, and explainable recommender systems. Commencing with general recommender systems, the discussion begins with traditional recommendation approaches. Subsequently, more advanced recommender systems including those utilizing deep learning techniques and HINs are discussed. The subsequent section introduces visually-aware recommender systems, along with an overview of their current state-of-the-art. It also addresses an issue pertaining to the explainability of visually-aware recommender systems. The last section focuses on explainable recommender systems. It first introduces the concepts of explainability in AI and examples of explainability in recommendations. It then explores the domain of explainable HIN-based recommender systems, particularly focusing on the methodologies used by state-of-the-art approaches to generate explainable recommendations. Furthermore, this section provides a summary of benchmark datasets commonly used in explainable HIN-based recommender systems, alongside evaluation methods to assess explainability. Figure 2.1 presents an overview of the technical development of various recommender systems in this literature review. Table 2.1 shows the timeline depicting the technical development of key traditional recommender systems, visually-aware recommender systems, explainable visually-aware recommender systems, and explainable heterogeneous information network (HIN)-based recommender systems. Further details on different types of recommender systems are provided in the following sections. By meticulously assessing the existing research approaches, this literature review ultimately identifies gaps in explainable visually-aware recommender systems, which subsequently leads to the goal of this thesis.

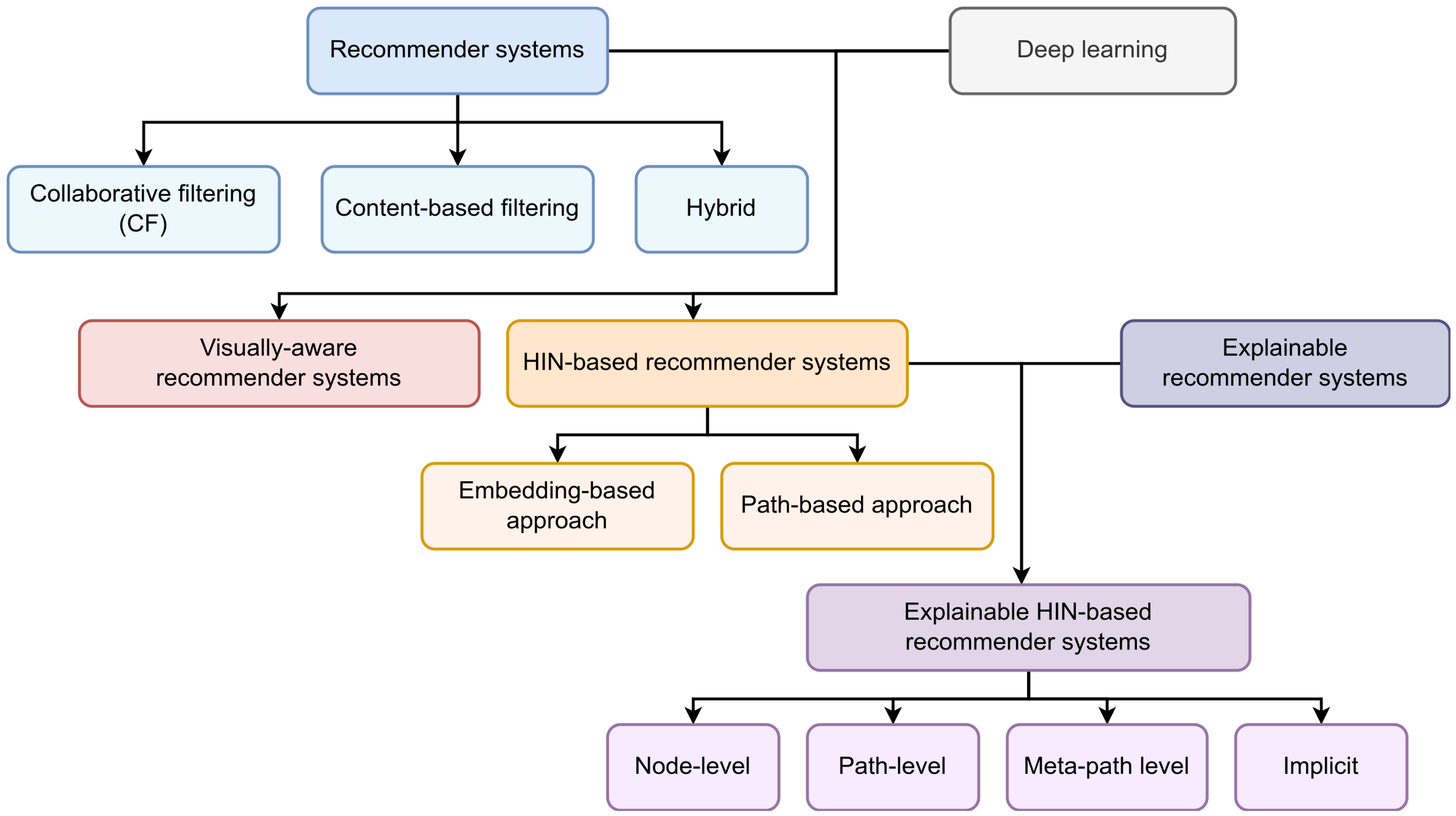

Recommender systems
Deep learning
Collaborative filtering (CF)
Content-based filtering
Hybrid
Visually-aware recommender systems
HIN-based recommender systems
Explainable recommender systems
Embedding-based approach
Path-based approach
Explainable HIN-based recommender systems
Node-level
Path-level
Meta-path level
Implicit

Figure 2.1: Overview of the technical development of various types of recommender systems in the literature review of this thesis

Table 2.1: Timeline depicting the technical development of key traditional recommender systems, visually-aware recommender systems, explainable visually-aware recommender systems, and explainable heterogeneous information network (HIN)-based recommender systems

	Traditional recommender system	Visually-aware recommender system	Explainable visually-aware recommender system	Explainable HIN-based recommender system
1994	One of the earliest collaborative filtering (CF) models [76]			
1997	One of the earliest content-based filtering models [77]			
2009	MF [78], BPR-MF [79]			
2016		VBPR [37]		
2017		DVBPR [38], DeepStyle [80], VPOI [81], CNN-PerMLP [82], Gasper's [83], JRL [84]		Metapath2vec [85]
2018				RippleNet [61], Knowledge-Aware Autoencoders [86], SEP [87], Ai et al.'s [70]
2019		aVBPR [41], Yu et.al.'s [88]	VECF [40], SEAR [45], Ante-RNN [89], MMalfM [43]	KGAT [5], PGPR [65], LDSDMF [90], KPRN [64], EIUM [91], RuleRec [69]
2020		CER [92]		HAGERec [62], MP4Rec [72], PRINCE [93], FairKG4Rec [94], CAFE [95], ADAC [67], MSRE [96]
2021				GEAPR [97], KGIN [98], TMER [73], LOGER [99], UCPR [100], MKRLN [101]
2022				KGAT+ [102], PLM-Rec [103], TPRec [104], ReMR [68]

2.1 Recommender Systems

Recommender systems are a type of artificial intelligence (AI) system used to suggest items, products, or services to users based on their preferences and behaviors. These systems are commonly used in online platforms such as e-commerce websites, social media networks, and content streaming services to help users discover new products or content that they may be interested in. Recommender systems can be generally categorized into three major approaches, i.e., a collaborative filtering approach that uses the behavior and preferences of similar users to make recommendations, a content-based filtering approach that recommends items based on the characteristics of the items themselves, and a hybrid approach that combines collaborative and content-based filtering approaches.

Collaborative filtering A collaborative filtering (CF) approach [76] predicts recommendations based on user-item historical information no matter it is explicit feedback (e.g. ratings) or implicit feedback (e.g. buy, tag, or watch). The main idea of this approach is that users who have interacted with the same items are likely to have similar preferences. A CF approach then identifies similar users with similar interests and makes recommendations on this basis. For example, if a user has rated several movies highly in the action genre, the system identifies other users who have also rated these movies highly and recommends other action movies watched by these like-minded users to the user. Figure 2.2 illustrates an example of how a collaborative filtering approach predicts a recommendation. In this figure, "User 1" is considered to be similar to "User 2" because they both like "Skyfall". Since "User 2" also likes "Inception", this movie can then be recommended to "User 1". Despite the effectiveness of a CF approach, the performance of this approach becomes poor when user-item interactions are insufficient. This is commonly known as the sparsity problem [14]. Generally, the number of items is much larger than the number of users, and users only interact with a small subset of the available items. This results in sparse user-item interaction data, where most of the data is missing or unknown. Another issue is the cold-start problem [13]. When there is a new user introduced to the system, it can be difficult to personalize his/her recommendations effectively since there is not enough information about his preferences. To address such problems, several techniques have been developed. One common technique is to use matrix factorization (MF) methods. These methods can handle sparse data by learning a low-dimensional representation of the data that captures the underlying structure of the user-item preferences. The main idea is to decompose a large and sparse user-item interaction matrix into two lower-dimensional

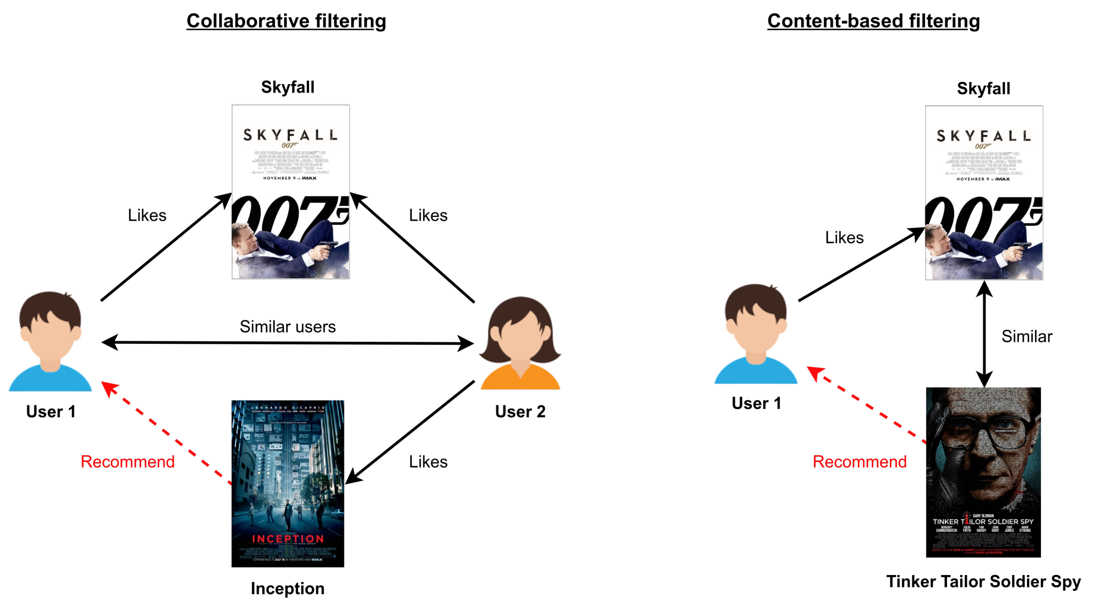


Figure 2.2: Example of how a collaborative filtering approach and a content-based filtering approach predict a recommendation. In this scenario, using a collaborative filtering approach, "User 1" is recommended "Inception" since "User 2" likes it and "User 1" and "User 2" share similar tastes stemming from their mutual appreciation for "Skyfall". On the other hand, using a content-based filtering approach, "Tinker Tailor Soldier Spy" is recommended to "User 1" due to its thematic similarity to "Skyfall".

matrices that capture the latent features of users and items, i.e., a user-latent-factor matrix and an item-latent-factor matrix. The number of latent factors of each user/item is typically much smaller than the original dimension of the user-item interaction matrix. Therefore, this method allows for higher computational efficiency and reduces the impact of sparsity. Once the user-latent-factor and item-latent-factor matrices have been computed, the predicted rating for a user-item pair can be obtained by computing the dot product of the corresponding row of the user-latent-factor matrix and the column of the item-latent-factor matrix. This predicted rating can then be used to generate personalized recommendations for each user by recommending items with the highest predicted ratings that the user has not yet interacted with.

MF methods use various techniques to learn the latent factors of users and items. The most commonly used algorithms are Singular Value Decomposition (SVD) [105], Non-negative Matrix Factorization (NMF) [106], and Alternating Least Squares (ALS) [107]. Bayesian inference can also be used to learn user and item latent factors. Bayesian Personalized Ranking

Matrix Factorization (BPR-MF) [79] is a method that combines MF and Bayesian inference to predict the likelihood of a user preferring one item over another. In BPR-MF, the user-item interaction matrix is decomposed by training the model to optimize a ranking objective instead of a prediction objective. The goal is to learn a ranking function that maximizes the probability of a user preferring an item over another, given their past interactions. To optimize the model's parameters, BPR-MF assumes a prior distribution over the parameters and computes the posterior distribution over the parameters given the observed data by using Bayesian inference. This allows the model to incorporate uncertainty and make probabilistic predictions. Moreover, BPR-MF is particularly useful for situations where explicit feedback (such as ratings or reviews) is not available and only implicit feedback (such as clicks, purchases, or views) can be used to learn about user preferences. Due to this advantage, it has been used in a variety of recommender systems.

Content-based filtering A content-based filtering approach [77] is based on comparisons of item metadata and user personal profiles. Given a user and his/her historical data, content-based filtering aims to recommend items that are similar to those that have been interacted with by this user before. In other words, when a user interacts with an item, the system uses its attributes to find other items with similar attributes and recommends them to the user. For example, in Figure 2.2, "User 1" likes "Skyfall". Since "Skyfall" and "Tinker Tailor Soldier Spy" share similar genres, "Tinker Tailor Soldier Spy" can then be recommended to "User 1". Since this approach does not rely on user-item interactions of other users, it is suitable when user-item interactions are sparse. However, this approach is limited to recommending items of similar characteristics as the ones the user has already interacted with. Other users' interests cannot be fully explored in order to make accurate and more diverse recommendations.

Typically, content-based filtering recommender systems use feature extraction techniques to generate item representations and calculate the similarity between items. Different types of data can be leveraged to generate representation vectors that capture the characteristics of the items. Metadata is one of the common data types that have been extensively used to produce recommendations [6]. Metadata refers to the data that describes an item, such as actors, directors, or released dates for movies. In some cases, this information can be used to generate item representations and calculate similarity scores between items. Textual data is also widely used in recommender systems. For instance, in movie recommender systems, the title, synopsis, and reviews of the movie can be used to extract keywords, themes, and genres. Content-based filtering models can also extract visual features from images of items to model item visual profiles. For example, in a fashion recommender system, the color, pattern, and style of clothing items can be considered when generating feature vectors that capture the visual characteristics of the items [37, 38, 40]. Several techniques can be used to generate item representations using any of these types of data. One of the common techniques is Vector Space Models (VSM) [108] which is a technique that represents items as vectors of their feature values. Each feature represents an attribute of the item such as keywords, topics, tags, categories, etc. VSM models weight each feature by its importance to the item and calculate the similarity between items using cosine similarity or Euclidean distance. Term Frequency-Inverse Document Frequency (TF-IDF) is popularly used to weight the importance of item features in the VSM. Originally, TF-IDF was proposed for Natural Language Processing (NLP) tasks. It is used to measure the importance of a term in a document and is ubiquitously applied in information retrieval systems and text mining. This method can be adapted to assign higher weights to features that are unique to an item and lower weights to features that are common across many items. Latent Semantic Analysis (LSA) is another common technique used in content-based recommender systems. This method aims to identify latent relationships between features in the item vectors. Specifically, it reduces the dimensionality of the item vectors and identifies the underlying semantic meaning of the features. This results in latent feature vectors with much lower dimensionality than the original feature vectors. Then, the similarity of their latent feature vectors is computed to finally produce recommendations. Rule-based systems generate recommendations by considering a pre-defined set of rules based on item attributes. These rules can be derived by using heuristic methods or prior domain knowledge to identify the most suitable items for a particular user. Deep learning models, such as neural networks, can also be used to generate item representations. These models can be trained on large amounts of data to extract significant features, learn relationships among features, and model effective item profiles for producing recommendations.

There are several advantages of a content-based filtering approach. First, it can provide recommendations for new users or items that have not yet been rated or interacted with, since it only relies on the features of the items rather than user behavior. Additionally, it can suggest niche or specialized items that may not have been interacted with by many users. However, one limitation of content-based filtering models is that they rely on the accuracy and completeness of the feature representation of items. In the case that these features do not capture the relevant characteristics of the item, the recommendations may be inaccurate. Furthermore, content-based filtering models may not account for changes in a user's preferences over time, which can affect the quality of recommendations.

Hybrid of CF and content-based filtering To overcome the disadvantages of CF and content-based filtering approaches, modern recommender systems have been using a hybrid approach that combines the strengths of both approaches. A collaborative filtering approach uses behaviors and preferences of similar users to recommend items to a target user, while a content-based filtering approach recommends items based on the features of the items themselves. By combining these two approaches, both users' behavior data, and item features can be simultaneously utilized. Hybrid recommender systems can be particularly useful when user-item interaction data is sparse or noisy, or when there is a high degree of item or user diversity. By leveraging multiple recommendation techniques, a hybrid system can generate more accurate and relevant recommendations for users. Nevertheless, as data becomes more complex and users' preferences show higher diversity, there is an increasing requirement for more proficient recommendation models. This is where deep learning models are involved in order to enhance the capabilities of hybrid recommender systems.

2.1.1 Deep-Learning Based Recommender Systems

Advancements in deep learning have created new opportunities for enhancing hybrid recommender systems. With advanced neural network architectures, deep-learning based recommender systems can effectively capture intricate patterns and relationships within high-volume and entangled data [9]. There are numerous neural network architectures that have been utilized to develop deep-learning based recommender systems. Examples of these architectures include autoencoders [109], Recurrent Neural Networks (RNNs) [110], Convolutional Neural Networks (CNNs) [28], and Deep Reinforcement Learning [111]. Autoencoders, for instance, are designed to learn efficient representations of input data by encoding it into a lower-dimensional space and then decoding it back to its original form. Within the context of recommender systems, autoencoders can be used to learn compact representations of users and items, which are subsequently leveraged for making personalized recommendations [86, 112, 113]. RNNs have been effectively utilized to capture the temporal dynamics of interactions and sequential patterns in user behaviors [114, 115]. Apart from this, RNNs have also proven to be valuable in utilizing textual reviews to enhance recommendations [116, 117]. Several recommender systems employ CNNs to extract features from images, which are then incorporated into learning users' preferences [37, 38, 81]. While CNNs are primarily associated with image data, they can be adapted and extended to handle different modalities in recommender systems including texts [118, 119], user-item interaction matrices [120], and sequences [121]. Deep Reinforcement Learning combines the ability of deep learning to model complex patterns in data and the principles of reinforcement learning to optimize decision-making processes. In recommender systems based on Deep Reinforcement Learning, the recommendation process is treated as an interaction between an agent (the recommender system) and an environment (the user or item space) [65, 122, 123, 124]. The agent learns to provide recommendations by selecting actions that maximize a predefined reward signal.

As evident from these prior studies, deep-learning based recommender systems can capture more intricate user preferences and item characteristics. They allow for the incorporation of auxiliary information from images, texts, and other types of data to better model users' preferences and capture item characteristics. This enhances the quality and accuracy of recommendations, as it considers a broader range of user behaviors and item attributes.

2.1.2 HIN-Based Recommender Systems

Deep learning has also brought another significant breakthrough by incorporating HINs into recommender systems and given rise to another recommendation approach known as HIN-based recommender systems. These systems utilize high-order connectivity in HINs to learn users' preferences and the potential connections between users and items that they have not yet interacted with. This connectivity information is not only beneficial for modeling users' profiles and predicting recommendations but also for providing explanations of recommendations. Due to this advantage, recommender systems using HINs have become ubiquitous during the past decade [10, 60]. Based on different methods of extracting high-order connectivity information, HIN-based recommender systems can be divided into three approaches: embedding-based, path-based, and hybrid approaches.

Embedding-Based Approach An embedding-based approach uses node embeddings generated from network/graph embedding methods [125] to produce recommendations. For example, Bellini et al. [86] proposed a recommender system based on Semantics-Aware Autoencoders (SemAuto) that involve structural information in a network in an Autoencoder Neural Network. Each neuron in SemAuto represents a node in the HIN, and the weights on all neurons are learned to build user profile representations. By identifying the importance of each node towards a specific user, the node with the highest importance can be used as an explanation. Besides Autoencoder Neural Networks, translation-based methods, such as TransE [126] and TransR [127], are also utilized in some embedding-based models to create node embeddings. Given a network triplet (h, r, t), where h represents a head node, r represents a relation, and t represents a tail node, these methods assume that the combination of information from h and r should be equal to the information from t. This can be expressed as the equation $h + r = t$. Based on this assumption, the generated node embeddings can capture the network structure and can be effectively used to learn recommendations [70, 99].

Some embedding-based models use reinforcement learning (RL) to navigate paths that connect a user to his/her recommended item [68, 101, 104]. These models typically build a policy network to find the optimal path, maximizing the cumulative reward at the end of navigation. An agent starts from a given user and navigates through the network, moving from one node to the next adjacent node until it reaches the recommended item. However, navigating through the nodes and relations in a network can be computationally expensive. This may cause serious scalability issues for RL-based models. In [65], a user-conditional action pruning strategy was proposed to decrease the size of the action space and reduce the computational cost of their RL-based model. In [101], an attention mechanism was employed to weigh the importance of each neighbor, allowing the agent to focus on the most relevant actions. These attention weights guide the agent in reducing the number of possible action spaces. Another potential problem with RL-based models is that the agent guided by the policy network may repeatedly search the same path with the largest cumulative rewards, leading to recommendations that lack diversity.

Another line of embedding-based models utilizes Graph Neural Networks (GNNs) to propagate multi-hop information recursively. These GNN models update each node embedding by considering the embeddings of its neighbors, allowing them to capture high-order connectivity. One of the state-of-the-art models is the Knowledge Graph Attention Network (KGAT) [5] which combines a Graph Convolution Network (GCN) and a Graph Attention Network [128]. In this model, TransR was first used to generate node embeddings. These embeddings were then refined by recursively propagating their neighbors' embeddings. A knowledge-aware attention mechanism was employed during propagation to identify important neighborhoods. Despite being effective, KGAT can be computationally expensive when the number of nodes and relations is large. An improved framework called KGAT+ [102] was proposed to address the issue of scalability in GNN-based models. Multiple one-to-many relations were compressed by using a latent class model. Specifically, a set of target relations r was first defined. Next, for any triplet (h, r, t), a latent class between h and t was assumed. The relation between h and t was decomposed into two relations: h and the latent class and t and the latent class with specific probabilities. These probabilities were then taken into consideration when computing attention weights and loss for learning recommendations. To better profile users and capture their interests, some GNN-based models incorporate higher-level concepts beyond nodes and relations in HINs. For instance, the Knowledge HIN-based Intent Network (KGIN) [98] was proposed to model users' underlying intents which are higher-level information. Each intent was formed by attentively combining relation embeddings in a network. In [129], a modification to the traditional GCN model was proposed to factorize user and item embeddings based on multiple concepts. To achieve this, the aggregation at the last layer was adjusted to distinguish information from the lower layers based on concepts. Specifically, all embeddings from the lower layer were classified into different concepts and then propagated separately. To categorize the embeddings into different concepts, the factor affiliation degrees were estimated from the learnable parameters of each factor.

Path-Based Approach Given a user and his/her item, paths connecting them in a HIN provide valuable information to learn about his/her preferences. A path-based approach extracts these paths from a HIN and feeds them to a recommendation learning framework. Depending on the techniques used for path extraction, a path-based approach can be categorized into two groups. The first group is random-walk based models, which extract or sample paths randomly to learn recommendations. Additional biases or conditions, such as relation weights and restarting strategy, may be added to ensure the quality of the sampled paths. The second category is meta-path based models, which use paths extracted via predefined meta-paths to feed the recommendation framework. These extracted paths are more controllable and semantically meaningful compared to randomly sampled paths. They can also provide explainable connectivity information between users and items.

Random-walk based models utilize random walk [130] to extract paths from a HIN. This method starts from a given node and randomly selects an adjacent node to form a path until a predetermined length is reached or certain conditions are satisfied. In [64], an LSTM network was applied to capture the sequential dependencies of nodes and relations within the randomly sampled paths. Since different paths may have different impacts on the user-item connectivity, a weighted pooling method was adopted to combine all path representations. Then, based on the aggregated paths representation, the recommendation score for the user-item pair was calculated using fully-connected layers. Inspired by a concept of an auto-regressive path language model [131], randomly sampled paths were used to build a path language model for recommendations in [103]. After building the model, a set of candidate path instances that connect a user node to a recommended item node were generated using a Transformer-based decoder. The recommendation scores were determined by calculating the joint probabilities of these candidate path instances. One potential issue with using random walks is a lack of diversity in extracted paths. When using random walks to extract paths, nodes with higher degrees are more likely to appear in the paths than nodes with lower degrees. This can result in a lack of diversity in the extracted paths and biased recommendations.

The connectivity information obtained from random-walk based models is typically implicit and unpredictable since the paths are randomly extracted. To obtain meaningful connectivity information, meta-path based random walk was introduced [71]. Instead of randomly selecting the next node from the neighborhood, meta-path based random walk selects the next node in accordance with the node type specified in a meta-path at each step. This approach allows for obtaining high-order connectivity under specific assumptions and can be utilized for learning recommendations. In [73], item-to-item meta-paths were also used to extract path instances. These path instances were then used in Word2Vec [132] method to generate path embedding. To enrich item embedding with reasoning information from the path instances, a self-attention mechanism was employed to combine the path embedding with item embedding. This process was repeated for all consecutive item pairs. The obtained item embeddings, user embeddings, and path embeddings were utilized in the final step to predict recommendation scores. In [96], two structural relations, i.e., affiliation relation (one-centered-by-another structure) and interaction relation (peer-to-peer structure) were leveraged by using different types of meta-paths. For affiliation relation, the meta-paths following these two patterns namely user-item-features and item-user-attributes were used. For interaction relation, meta-paths starting with a user node type and ending with an item node type were used. Since there could be a number of possible meta-paths following these formats, a set of meta-paths was sampled based on the computed association degrees. Once the meta-paths were selected, meta-path embeddings were computed and attentively aggregated using an attention mechanism. Meta-paths can also be used to generate user/item features that capture multi-hop information in HINs. In [69], given a user's item and an item of interest, item-to-item meta-paths were selected based on predefined criteria. Based on these chosen meta-paths, the probabilities of finding path instances between the item pair following these meta-paths were then computed. These probabilities were used to generate feature vectors of item pairs. Such feature vectors were subsequently used to learn recommendations in the BPR-MF model and NCF model. In [72], PathSim [133] and metapath2vec [85] were utilized to construct user-user and item-item similarity matrices based on various meta-paths. For each meta-path, a Multi-Layer Perceptron (MLP) model was trained on the corresponding similarity matrix to learn user and item latent factors. An attention mechanism was then applied to combine all latent factors from each meta-path to obtain the final user/item latent factors.

In general, the path-based approach leverages multi-hop relations to model users' preferences by using paths extracted from HINs. Accordingly, the performance of this approach depends highly on the quality of these extracted paths. Extracted paths with noise may reduce the effectiveness of learning users' preferences. To mitigate this problem, extracting paths at a certain length while ignoring remote connections allows for learning recommendations effectively [133, 134]. Moreover, the process of extracting paths from a HIN can be time-consuming since the number of possible paths connecting two nodes grows exponentially when the number of nodes and relations increases [122]. Developing a path-based model that is scalable remains a challenge in this research field.

Hybrid Approach The embedding-based approach uses node embedding techniques, which offer high flexibility and efficiency compared to the path-based approach. The path-based approach, on the other hand, utilizes multi-hop relations from HINs in an explicit and intuitive manner. The hybrid approach combines the strengths of both the embedding-based and path-based approaches to leverage the overall HIN structure and extracted paths. RippleNet [61], an end-to-end hybrid model, uses the concept of ripple sets to extract multi-hop relations. A ripple set is a set of triplets within n hops from an item node of a given user. In their work, ripple sets of different n for a user and one of their previously interacted items were first generated. Starting from the 1-hop ripple set, information from each triplet in the set was propagated based on relevance probability. This probability was computed from the similarity between the item embedding and the head node embedding of each triplet. The user embedding was computed by the weighted sum of the tail node embeddings in the ripple set, based on these relevance probabilities. This process was repeated until reaching the n-hop ripple sets, resulting in the final user embedding enriched with multi-hop information. The user and item embeddings were then used to predict the recommendation score. In [91], textual information from user reviews, and visual information from item images were jointly utilized with multi-hop relations. To incorporate these different modal features, additional constraints based on these features were added to TransE. Based on the learned node embeddings from this constrained TransE, each user-item interaction representation was computed by attentively combining path instances connecting them. All user-item interaction representations were aggregated to predict recommendations. A multi-modal hybrid model was also proposed in [97]. Structural context and neighbor impact from HINs, user attributes, and geolocation influence were jointly considered in their work. To aggregate the multi-modal factors, a triple attention mechanism was adopted. To leverage the structural context of a user, random walk with restart (RWR) and a multi-layer neural network were adopted to learn the structural context latent factors. For the user's neighbor impact, an attention-based HIN neural network was used to aggregate information from their neighbors and identify the most significant neighbor based on the attention weights. Latent factors of user attributes, categorical and numerical features, were generated using an attention-based latent factorization machine. All these multi-modal features were finally combined using an attention mechanism strategy.

In [95], a hybrid model for generating paths connecting users to their recommended items was proposed. In their model, user profiles were considered as coarse sketches of user behaviors, using user-centric patterns that reflect the user's interests. To find these patterns, an off-the-shelf random-walk based algorithm was adopted to identify a set of candidate user-centric patterns for each user. Then, the prominence scores of selected user-centric patterns were computed to indicate their likelihood of generating paths from the user to potentially recommended items. Based on the obtained user profile, a Profile-guided Path-Reasoning algorithm (PPR) was introduced to find a collection of paths using selective neural symbolic reasoning modules. Compared to RL-based models, their model is more efficient since it can collectively find multiple paths instead of finding each path separately. In [135], a GCN was adopted to learn node representations by applying a weighted sum aggregator to capture the features of each node and its neighbors. Apart from this, a heuristic path search algorithm was also used to extract path instances that represent a user's potential interests. The selected paths along with the node representations were then fed into an LSTM model with a self-attention mechanism to encode the sequential dependencies of the nodes and produce an encoded vector. Finally, multiple MLP layers with an activation function were applied to the encoded vector to predict the recommendation score.

As can be seen in these previous studies, HINs have become highly beneficial for learning accurate recommendations. Apart from accuracy, HIN-based recommender systems yield another pivotal advantage which is offering explainability in recommendations. Unlike traditional recommendation systems or other deep-learning based recommender systems that frequently function in a black-box manner, HIN-based recommender systems can provide the rationale behind each recommendation. They leverage the structure and relationships present in HINs to provide transparent and interpretable recommendations. These systems enable users to comprehend the rationale behind the recommended items. This increased transparency not only enhances user trust but also allows for better fine-tuning of recommendations based on individual preferences.

Despite the success of existing HIN-based recommender systems in terms of both accuracy and explainability, they only consider and leverage semantic information such as user item metadata in HINs. Visual information from images is also another type of information that can be valuable in recommendations. Including such information could highly improve the accuracy of recommender systems. In spite of that, most existing studies typically ignore visual information. As a result, there is still a gap in augmenting visual information in HINs for learning recommendations. By filling this gap, HINs can be used to facilitate the task of visually-aware recommendations as well as introduce explainability in these recommendations using high-order relationships within these networks.

2.2 Visually-Aware Recommender Systems

Traditional recommender systems use various techniques to identify user preferences and recommend relevant items. However, traditional recommender systems may not be effective for products with visual features, such as clothing or furniture, where users are often interested in specific styles, colors, or shapes. Visually-aware recommender systems are a class of recommender systems that take into account visual information in addition to user-item interaction data. These systems leverage visual features such as colors or shapes of items extracted from images, videos, or other types of visual content. They analyze both user-item interactions and these visual features to recommend items that are visually appealing to users. For example, a visually-aware recommender system for clothing may recommend items based on the user's preferences for specific styles, patterns, or colors. Owing to advances in computer vision and image processing, visual features can be extracted by using several feature extraction methods [25, 26, 27] or deep learning models such as convolutional neural networks (CNNs) [136].

These features have been proven to be highly useful for representing visual information of item images. They can be integrated into recommender systems as additional information along with user-item interactions to learn users' preferences more effectively.

One of the firstly proposed visually-aware recommender systems is the Visual Bayesian Personalized Ranking model (VBPR) [37], a modified BPR-MF model that incorporates visual information into learning user/item latent factors. Novel user/item latent factors subject to visual preferences were introduced and learned by projecting image features to the visual latent space using an embedding kernel. This embedding kernel is a learnable weight matrix that linearly transforms a high-dimensional item image feature to a lower-dimensional space, i.e., the visual latent space. The image features were extracted by using the Caffe reference model pre-trained on 1.2 million ImageNet (ILSVRC2010) images [136]. The output of the second fully-connected layer was chosen as an image feature vector representing the whole visual appearance of each image. Their work has also shown that, in their proposed visual space, visual preferences can be recognized as clusters of latent vectors in the space. To better model users' visual preferences, the modified version of VBPR called DVBPR was proposed in [38]. Instead of using a pre-trained model to extract visual feature vectors with an embedding kernel, visual features were learned by a CNN module simultaneously with learning recommendations. Specifically, the learnable weight matrix in VBPR was replaced by a deep learning module to jointly learn recommendations and visual features simultaneously. This module allowed DVBPR to model more complex visual preferences compared to VBPR. Moreover, DVBPR can also be combined with Generative Adversarial Networks (GANs) to synthesize item images depending on the user's preferences. Given a user and an item category, this system can generate new images of clothing items that potentially match with user's preferences. In [41], three additional models were integrated into VBPR to build an advanced Visual Bayesian Personalized Ranking (aVBPR) model. The first model is the Factorized Personalized Markov Chains (FPMC) model, used to simulate users' sequential behaviors. The second model is the intelligent Field-aware Factorization Machine (iFFM) model which was modified to predict users' preferences based on non-visual features. The third model is the VBPR model which was used for modeling users' visual preferences.

Apart from fashion items, visual information was also used to enhance other types of recommendations such as point-of-interest (POI) recommendations and tag recommendations. In [81], a POI recommendation framework using visual information called Visual Content Enhanced POI recommendation (VPOI) was proposed. In their work, visual features were extracted by using VGG16 [137], a CNN model pre-trained on ImageNet for image classification. They were used to guide the learning process of both user and item latent vectors. Given a user, a location, and an image of this location posted by this user, it was assumed that the user and item latent vectors should be able to differentiate if the image is associated with the user and the location subject to the extracted visual features. Based on this assumption, additional constraints were then integrated into Probabilistic Matrix Factorization (PMF) [138] to force user latent vectors and item latent vectors to be similar to the corresponding visual features. In [82], CNN-PerMLP was proposed to consider visual information in personalized tag recommendations. In this model, the visual features of each item image were extracted from multiple small patches within this image by using a pre-trained CNN model. Then, these visual features were personalized to each user by combining them with his/her personal information. After combining, latent features of an item image according to a specific user were obtained and used in a BPR framework to learn recommendations. In [83], visual features extracted from movie posters were compared with semantic features (metadata) for generating movie recommendations. Two types of visual features were considered, i.e., low-level features (e.g., based on the color characteristics) and high-level features (e.g., based on the content of the image). As for low-level features, image color features in the Hue-Saturation-Value (HSV) color space were considered. It has been shown that this color space represents an image in a similar way to human visual perception [139, 140]. For each of these color dimensions (i.e., hue, saturation, value), three features were computed, i.e., the mean value of all pixels, the standard deviation of all pixels, and the mean value of pixels on both diagonals. The Pleasure-Arousal-Dominance model [141] was also adopted to estimate the impact of six basic colors (i.e., red, green, blue, orange, yellow, and violet) on user emotions based on a combination of pleasure, arousal, and dominance. High-level visual features were extracted using Inception-v3 [142] which is the pre-trained CNN model by Google trained on the ImageNet dataset. These features were applied in a content-based recommendation approach. First, the Okapi BM25 ranking function [143] was used to generate a list of candidate items. These candidate items were then re-ranked based on the visual features extracted. In [80], the style features were introduced and used in the model called DeepStyle. According to their observation, items with similar styles may be not similar in the visual space of VBPR. One reason is that categorical information is more dominant in this common visual space than style information. Thus, categorical information was eliminated in their work by subtracting items' latent categorical representations from the visual features generated by a pre-trained

CNN model. Assume that a visual feature extracted from a pre-trained CNN model is a combination of style and category features (i.e., item = style + category). To consider only the style of each item, the style feature of each item was computed by subtracting its categorical feature from its visual feature (i.e., style = item − category). To learn recommendations, these style features were used similarly to the visual features in VBPR. These studies have shown the possibilities of using visual features extracted via several techniques to enhance recommendation performance. However, in some cases, using visual features extracted directly may not be sufficiently effective to capture users' preferences.

To further improve recommendation accuracy, images can also be simultaneously leveraged along with other types of data. Many researchers have been exploring techniques such as multi-modal learning, where deep neural networks are designed to jointly model visual features and metadata. These techniques aim to capture the interactions and dependencies between different modalities, enabling a more comprehensive understanding of user preferences.

In [84], review texts, item images, and user ratings were adopted to learn user and item representations. In their framework, information existing in each modality was embedded by using different deep learning models. For textual reviews, the PV-DBOW model [144] was adopted to learn review representations. For item images, the pre-trined CNN model as in [37] was adopted to extract initial image features from item images. These extracted features were then fed to a deep learning module to generate final image representations for learning recommendations. Representations from different modalities were subsequently integrated to obtain joint representations for both users and items. Finally, these integrated representations were used to train the recommender system in a unified pair-wise learning-to-rank framework. In [92], multimodal features, including audio, image, and motion features, were incorporated with user-video interactions to learn video recommendations. Two types of visual features were considered in their work, i.e., SIFT and the features extracted from the pre-trained CNN model from the VGG group [137]. For SIFT, OSIFT (opponent SIFT) [145] and MoSIFT (motion SIFT) [146], two variants of SIFT, were adopted to capture the scene and motion information in the videos respectively. OSIFT considers the light color change and shifting on the RGB color space to capture more robust scene contents. Meanwhile, MoSIFT uses the optical flow between frames to capture the motion contents in the video frames. As for CNN features, the pre-trained CNN model [137] was adopted to extract features from the 5th pooling layer with spatial pooling [147]. Both SIFT features and CNN features were applied in a collaborative embedding regression (CER) model which incorporates collaborative filtering with content features. Traditionally, this model can only consider single content features. To incorporate multiple content features from different resources, three feature fusion methods were used. The first method is to concatenate multiple content features forming single combined context features. The second method is using content latent vectors as in CKE [148]. However, since constructing a shared latent space among multimodal features typically leads to significant information loss, they proposed the third method which is aggregating prediction scores from multiple content features to form more accurate final scores. In [88], an interactive recommender system that adjusts the list of recommendations according to natural language feedback on the visual appearances was proposed. This system consists of two components: the visual dialog encoder and the visual dialog augmented cascading bandit. The visual dialog encoder was used to encode an input image by using the ResNet101 model pre-trained on ImageNet [149] and user textual feedback as in [150]. The encoded image and words were further concatenated and used as input of GRU followed by a linear mapping to generate a final encoding of an image-feedback pair. To learn recommendations, the traditional cascading bandit algorithm [151] was modified to incorporate textual input and visual features from the encoder along with user-item interactions. In [43], both visual features from item images and textual features from users' reviews were utilized to model aspect-aware users' preferences and items' properties. They made an assumption that different users have interests in particular aspects of items. Also, even for the same aspect, different users have different preferences based on that aspect. As a result, a predicted rating of any user-item pair should depend on the user satisfaction with each aspect of that item and the importance of that aspect to the user. Based on this assumption, a multi-modal aspect-aware topic model (MATM) was proposed to model the latent factors of each user's preferences on a specific aspect and the characteristics of an item on that aspect. In their work, each aspect was represented as a distribution of the same set of latent topics. For each user and each item, the corresponding aspect was computed based on the weighted sum of these latent topics. An importance score of each latent topic was determined by two factors, i.e., the textual reviews of this user towards this item and the visual features of this item image. To extract visual features, each image was divided into patches regarded as "visual words". To determine visual words, the visual features of each pattern were first extracted by using ResNet-152. Then, the K-means method was adopted to find K clusters of these features. Each cluster center was then treated as a visual term. Finally, for each patch of a given image, its nearest center was identified and then this patch was regarded as the corresponding visual term of this center.

Thus, an image can be represented by visual words which are a collection of visual terms.

These studies have demonstrated the benefits of incorporating visual information in recommender systems. Visual information provides a richer representation of items, capturing their visual aesthetics, style, and context. By considering visual features, recommendation algorithms can account for the visual preferences of users, leading to more personalized and visually appealing recommendations. However, as the majority of visually-aware recommender systems depend on the extraction of latent visual features, they frequently lack transparency and explainability. This insufficiency has the potential to reduce users' confidence and increase their skepticism concerning the reliability of these systems. Hence, enhancing the transparency of visually-aware recommender systems is unquestionably vital [16].

2.3 Explainable Recommender Systems

Most AI systems use machine learning or deep learning models to mimic the way a human brain works to make decisions on a given task. Such a process of mimicking is generally complicated and tends to be hidden underneath the model algorithms. As a result, these models are often referred to as black-box models [48] in which their decision-making mechanisms are concealed. During the early era of AI, due to the high performances of these black-box models, the problem of understanding how they work was not a concern for both developers and end users. However, recent research has shed some light on potential issues that might be caused by using such models without comprehension of how they generate outputs [47]. These issues include biases in making decisions or ethical violations which can cause serious consequences, especially in high-stake domains such as financial or medical domains [48]. To address such issues, some recent regulations such as the General Data Protection Regulation (GDPR) of the European Union and other countries [49] have been established. This emphasizes the importance of developing explainable AI for ensuring fairness and trust for both users and developers of AI systems.

Explainable AI can provide explanations or allow a human to understand the logic behind its decision-making process. In response to the explainability regulations, research in explainable AI has emerged and has been continuously receiving a large amount of attention. In this research area, there are two terms that are usually mentioned, i.e., "explainability" and "interpretability". According to the previous work [47, 152], the term explainability refers to the ability to provide an explanation of why a decision is made. Meanwhile, the term interpretability refers to the characteristic of an AI system in which its internal mechanism is comprehensible for a human. In the case of interpretable models, developers or users may take advantage of their interpretability to extract some explanations. Thus, whether an AI system has explainability or interpretability, it is possible to obtain an explanation from it.

Due to the recent requirements of explainable AI, explainability in recommendations is also required for modern recommender systems. Explainable recommender systems are capable of recommending items along with providing explanations of how or why those items are recommended. They provide not only transparency and trustworthiness of recommender systems but also persuasiveness to end users. With explanations, the recommendations can be more convincing to the users [10]. Besides end users, explanations also benefit developers or designers of recommender systems. They allow ease of diagnosing, maintaining, and refining recommender systems. These explanations can be in various formats depending on the types of resources or information used in recommender systems. For CF-based recommendation models, since these models use only user-item interactions, user-based and item-based explanations are two common types of explanations. These explanations are typically in the form of similar users or similar items [10]. However, in case user-item interactions are sparse, it can be difficult to obtain such explanations. As additional information has been leveraged to improve recommendation performances, it can also be used to provide explanations. Textual explanations are common in those recommendation models that use texts as additional information. These explanations are often in the form of text segments extracted from users' reviews [40, 43] or item descriptions [70]. Although textual explanations are easy to understand, there is a limitation when users' reviews or item descriptions are unavailable or not thorough enough to cover users' preferences. Also, users' reviews are typically noisy [10]. Not all of the sentences in a review are justifications for the users' decisions. Item images are another type of side information used for improving recommendation accuracy and explainability [40, 45, 89]. Typically, an image comprises various features, e.g., shapes, textures, and colors, which can be used for capturing users' preferences. Recent image-based recommender systems are capable of providing explanations in various formats such as highlighted regions in item images [40, 45] and highlighted regions with corresponding keywords derived from item images [89].

2.3.1 Explainable HIN-Based Recommender Systems

HINs offer a unique advantage to enhance recommendation accuracy and explainability. Unlike traditional recommender systems that often provide recommendations based on straightforward associations between items and user preferences, HINs capture intricate multi-hop relations between diverse entities. These high-order relationships can span multiple types of nodes and relations. They represent complex patterns of interactions and offer a transparent narrative explaining why a particular recommendation is made. With this advantage, explainable HIN-based recommender systems have become ubiquitous. Such systems provide a structured framework for extracting or harnessing multi-hop relations that connect users and items through intermediate entities. Generally, there are different levels or types of explainability that can be obtained from explainable HIN-based recommender systems. These levels include:

- node level, e.g., predictive/significant nodes, counterfactual nodes, or node neighborhoods
- path level, e.g., selected paths connecting users and recommended items
- meta-path level, e.g., predictive/significant meta-paths
- implicit level which means that explicit explanations are not generated along with the recommendations but the recommendations with higher explainability are encouraged to be produced rather than those with lower explainability

Node Level Node-level explanations include explanations that are in the form of nodes or 1-hop relations. These explanations are the simplest explanation type for explainable HIN-based recommendation models. Some models aim to identify certain nodes/relations that are influential or important for the recommendations of a given user. For example, SemAuto [86] treats the neuron weights in Autoencoder Neural Network as the importance weights of nodes in a corresponding network. Then, it identifies the most important node connected to a given user and uses it as an explanation. GEAPR [97] uses a GNN model with an attention mechanism to aggregate information among each user's neighborhood. Thus, the most important neighbor node for each user can be specified based on the attention weights. LDSDMF [90] identifies a node in a HIN that potentially matches a given user's preferences based on the likability degree computed. KGIN [98] models user-item interaction intents based on the relations connecting them in a HIN. Such relations are combined via an attention mechanism to form an intent representation. The relation with the highest attention weight is treated as an explanation in this model. Apart from important nodes/relations, node-level explanations can also be nodes/relations that are counterfactual conditions. For instance, PRINCE [93] provides explanations in the form of relations that, if they are removed from the network, will change the recommendation results.

Path Level Given a user node and an item node in a HIN, a sequence of nodes and relations that interlink them forms a path, offering comprehensive high-order insights into their connectivity. This connectivity can encompass user-user, item-item, or user-item relationships, among others. These paths are particularly suitable for serving as explanations of the reasoning behind recommendations—clarifying how a user is linked with their recommended item. As paths containing high-order connectivity information are often extracted or considered to model user profiles and facilitate recommendation learning within HIN-based recommender systems, using them as explanations becomes an intuitive and direct approach. As a result, numerous explainable HIN-based recommender systems have been focusing on providing path-level explanations. SEP [87] and FairKG4Rec [94] extract candidate paths connecting a given user and his/her recommended item in a HIN and rank them based on certain metrics and conditions. Path-level explanations can also be modified or rewritten to obtain more user-friendly explanations than sequences of nodes and relations. In the model proposed by Ai et al. [70], candidate paths are ranked by their probabilities based on the distribution of node embeddings learned by TransE. Then, the path with the highest probability is selected and applied on a pre-defined template in natural languages. Besides defining criteria or metrics to measure the significance of candidate paths, some models such as KPRN [64], EIUM [91] and TMER [73] use attention mechanisms to attentively combine multiple paths and identify the most significant one from the attention weights. This identified path is then used as an explanation path for the recommendation. KGAT [5], KGAT+ [102], HAGERec [62] use attention mechanisms to determine the importance of each node. For a given user and his/her recommended item, an explanation path connecting them can be formed by considering the attention weights in a cascading manner. RippleNet [61] similarly leverages the relevance probabilities to form an explanation path connecting a user and his/her recommended items. Explanation paths can also be generated based on the distribution of nodes in a HIN as in an auto-regressive path language model in which sentences are formed based on the distribution of words in a corpus. LOGER [99] uses an LSTM-based model to generate explanation paths.

PLM-Rec [103] generates explanation paths by using a Transformer-based decoder. CAFE [95] uses multiple neural symbolic modules to first generate a tree where the root node is a user node and the leaf nodes are the recommended item nodes. Based on this tree, multiple explanation paths can be extracted by tree traversal and ranked to find the most suitable one. PGPR [65], ADAC [67], UCPR [100], MKRLN [101], TPRec [104] and ReMR [68] formulate a recommendation task as deterministic MDP of path reasoning. The task is to navigate from a user node to his/her recommended item node in a HIN. Each and every navigation decision at each step altogether forms a leading path from this user to his/her recommended item which can be treated as an explanation for this recommendation. One advantage of these models is that paths discovered by a path reasoning process definitely exist in a HIN since they are generated by navigating through an actual HIN. This is different from those paths generated from node distribution where their explanation paths are formed based on the probabilities of node connectivity. Thus, there is no guarantee whether these paths exist in a HIN or not.

Meta-Path Level Path-level explanations provide explainability at an instance-level view. In other words, they particularly provide explanations in the forms of actual nodes and relations. On the other hand, meta-path level explanations provide explanations from a higher point of view, i.e., meta-level view. These explanations are in the form of node types or relation types in HINs. They can be used to explain recommendations based on the semantic meanings of these meta-paths. MSRE [96] leverages several meta-paths with different types to extract multi-hop relations. A meta-path embedding method aggregates information from these meta-paths by using an attention mechanism. The most important meta-path then can be identified based on the attention weights and used for explaining the recommendations. Similarly, MP4Rec [72] models user and item latent factors based on multiple meta-paths. For each meta-path, the initial user/item latent factors are learned based on the user-user/item-item similarity matrices based on each meta-path. Then, these user/item latent factors based on each meta-path are attentively combined. The most significant meta-path can be identified from the attention weights. Ma et al. [69] proposed RuleRec which is an explainable recommendation model based on the meta-paths derived from item association in a HIN. Given a user's item and an item of interest, this model finds the item-to-item meta-paths that reflect the association of these items. To find such meta-paths, hard-selection and soft-selection methods were proposed. Hard-selection method uses pre-defined hyper-parameters to validate meta-paths while the soft-selection method learns to select meta-paths with the additional objective function along with the recommendation objective function. After selecting the meta-paths, the probabilities of finding path instances between the item pair following these meta-paths were computed. These probabilities were used to create feature vectors of item pairs where each entry of this vector is the probability of each meta-path. Then, such feature vectors were used to learn recommendations in the BPR-MF model and NCF model.

Implicit Explanations that are at the node level, path level, or meta-path level are considered explicit explainability of recommender systems. However, instead of producing explicit explanations, some models provide explainability implicitly by selectively recommending items that are explainable rather than non-explainable ones. In [72], the objective function of a BPR framework was modified to increase the chance of generating more explainable items rather than those non-explainable ones. Specifically, an additional term was included in the classic BPR loss function. This term constrains the distance of the user and item latent factors in the latent space to be close to each other (i.e., their difference is close to zero) if the item is explainable to the user. The more they are explainable, the closer their latent factors are in the latent space.

2.3.2 Benchmark Datasets

Explainable HIN-based recommender systems have been applied to various kinds of datasets ranging from a movie domain to a retail domain. Some benchmark datasets that have been commonly used in the previous work are shown in Table 2.2. The details of these datasets are as follows:

- **MovieLens** [153] This dataset is a widely used benchmark dataset in movie recommendations. It contains user records of both explicit feedback (rating) and implicit feedback (watching and tagging). Regarding items, this dataset contains metadata of movies such as genres, directors, actors, tags, etc. There are multiple versions of this dataset with different scales including MovieLens-HetRec-2011, Movielens-100K, Movielens-1M, and Movielens-20M. To incorporate more rich information about movies, the combination of Movielens-1M and IMDb datasets linked by the tiles and release dates was used in KPRN [64]. In [90], the MovieLens-100K dataset was combined with SPARQL, a semantic web query language. In [91], the Freebase data dumps [154] was used to construct the HIN. The mapping relationship between movies of MovieLens-20M and entities of Freebase was adopted from [155, 156]. In [100], Microsoft Satori [157], a

Table 2.2: Benchmark datasets used in state-of-the-art explainable HIN-based recommender systems

Dataset	Variation	Model
Movielens	Movielens-100K	[72, 87, 90]
	Movielens-1M	[61, 62, 87, 100]
	Movielens-20M	[62, 86, 91]
Amazon	All categories	[93]
	Automotive	[73, 99, 103]
	Beauty	[65, 67, 68, 70, 94, 95, 100, 104]
	Book	[5, 98, 100, 135]
	CDs and Vinyl	[65, 70, 94, 95]
	Cell Phones and Accessories	[65, 67, 68, 69, 70, 94, 95, 99, 100, 103, 104]
	Clothing Shoes and Jewelry	[65, 67, 68, 70, 94, 95, 100, 104]
	Electronics	[69, 72]
	Grocery	[99, 103]
	Musical Instruments	[73]
	Toys and Games	[73]
Last.fm		[5, 62, 98, 129, 135]
Yelp		[5, 72, 96, 97, 129, 135]
Book-Crossing		[61, 62, 101]
KKBOX		[64, 101]

knowledge base developed by Microsoft, was adopted. This base uses the Resource Description Framework and the SPARQL query language and consists of billions of entities. It was used to build the HIN where the confidence level was set to greater than 0.9.

- **Amazon** [158] This dataset contains product reviews and metadata from Amazon which include 142.8 million reviews spanning May 1996 to July 2014 from around 20 million users. This dataset includes reviews (containing ratings, textual reviews, and helpfulness votes) and item metadata (descriptions, category information, price, brand, items that were bought/viewed together, and image features). In [93], the whole dataset with items from multiple categories was used in order to examine the effect of cross-category information on generating explanations. However, the whole dataset can be divided into smaller datasets of different product categories. Each dataset can be used as an individual benchmark which means that the results obtained from these divided datasets are not necessarily comparable to each other [95]. Some examples of these datasets of various categories along with the papers in which they were adopted can be found in Table 2.2. Similar to the MovieLens datasets, side information from publicly available knowledge bases can be incorporated to construct HINs for recommender systems. In [5] and [135], the HIN was constructed by mapping the book titles in the Amazon-Book dataset to the corresponding entities in Freebase [159]. However, the Amazon dataset can be highly sparse, several studies adopted the k-core setting where those users who have interacted with at least k items and items that have been interacted by at least k users were retained in the dataset.

- **Last.fm** [160] Last.fm is a music recommendation dataset extracted from Last.fm online music systems. It contains binary implicit feedback (tagging data) between users and music as well as social networking and music artist listening information. Also, similar to the Movielens and Amazon datasets, side information from Freebase can be incorporated to construct HINs as in [135].

- **Yelp** [161] This dataset offers comprehensive information about businesses and users on the Yelp website where users can rate local businesses or post photos and reviews about them. It provides user data consisting of information about each user's account, such as the account creation date, the number of reviews posted, the average rating given by the user, and the social network and friendship data among users. As for businesses, this dataset provides information about the business category, location, hours of operation, average rating, number of reviews, and the count of user check-in by date. In total, there are 1,017 categories of businesses (e.g., "Restaurants", "Bars", etc.) and 6,990,280 reviews over 150,346 businesses.

- **Book-Crossing** [162] his is a book dataset consisting of approximately 1,149,780 ratings (ranging from 0 to 10) from users in the Book-Crossing community. In [61] and [62], these ratings were converted to implicit feedback. However, due to the sparsity of this dataset, no rating threshold was set, i.e., each user-book interaction was marked with 1 if the user has rated the book regardless of its rating score.

- **KKBOX** [163] This music recommendation dataset was provided by a music streaming platform KKBOX. The KKBOX dataset consists of user-song interaction recorded within a specific time duration. For users, the metadata includes user ID, city, age, gender, registration method, registration date, and expiration date. For items (songs), the metadata includes song ID, song length, genre, artist, composer, lyricist, and language. In [101], the entities in the KKBox dataset were mapped to the CN-DBpedia [164] knowledge base [165] to construct the HIN.

However, benchmark datasets for evaluating explainability are not ubiquitous. There have been some efforts to generate test sets used for evaluating explainable recommendations or recommendation explanations obtained from HIN-based recommender systems. These test sets may provide guidance on how to construct benchmark datasets for explainable HIN-based recommender systems. In [67], test sets generated from users' reviews were used for evaluating the explanation paths obtained from their model ADAC. The idea is that an explanation path achieves high explainability if it contains many entities mentioned in the ground-truth review. To generate the ground-truth reviews, each user's review of his/her positive item was considered. For each review, certain words were filtered out if their frequencies were more than 5,000 or their TF-IDF scores were less than 1. The remaining words were treated as the ground-truth review for a comparison with the generated explanation path.

2.3.3 Explainability Evaluation

Evaluating explanations or explainable recommendations can be done in both qualitative and quantitative ways. A qualitative way usually involves data visualization or case studies to illustrate or examine how suitable explanations are. As for quantitative evaluation, several metrics have been proposed. However, since explainability in recommendations is relatively

Table 2.3: Qualitative methods adopted in state-of-the-art explainable HIN-based recommender systems

Qualitative method	Models
Visualisation	[97], [96], [129], [135]
Case study	[90], [129], [96], [98], [91], [103], [61], [95], [62], [5], [102], [73], [65], [101], [104], [68], [70], [64], [69], [94]
User survey	[86], [90], [100], [99], [135], [93]

a new area, there are not many common metrics for evaluating explanations or explainable recommendations.

2.3.3.1 Qualitative Evaluation

Most of the existing explainable HIN-based recommender systems evaluate the effectiveness of their explainability via qualitative evaluation. It can be a visualization of explanations or an analysis of case studies to decide whether their explanations are practical or not. Moreover, since generating explanations for users can be considered a human-computer interaction task, a user study can also be conducted to examine the quality of explanations generated by explainable HIN-based recommender systems. Table 2.3 summarizes the qualitative evaluation techniques that were used in some state-of-the-art explainable HIN-based recommender systems.

Visualization In [97], the authors demonstrated the interpretability of their model GEAPR by using heat maps of the attention weights showing the importance of features. These heat maps allowed them to know which features were informative for each input and indicated the lack of informative features in their experiments. Heat maps of attention weights on features were also used in [96] to evaluate the explanations generated for three randomly selected users. In [129], the item embeddings were visualized in scatter plots. In these plots, the color of each point indicated the closest affiliated factor of the item. They allowed the authors to observe the clusters of items based on the embeddings and investigate how the model discovered the factors and learned the embeddings correspondingly leading to high explainability. In [135], given a randomly selected user and his/her recommended item, the explanation paths with high attention scores were extracted to form reasoning sub-graphs. These sub-graphs were displayed to illustrate the explainability performance of their model. It can be seen that visualizing model structures (e.g., attention weights) or generated explanations allows users as well as model developers to understand the logic behind the decision-making processes. It provides a big picture of how the models work and whether the explanations generated are useful or not. Nevertheless, evaluation based on visualization can be subjective depending on observers. Thus, visualization might be suitable for an initial evaluation in which the overall performance of an evaluated model is generally examined. Then, it should be followed by further investigation using other evaluation techniques. Also, in the previous studies, some visualization may require sampling of users or items. This may not be sufficient to evaluate the actual explainability performance of those explainable HIN-based models.

Case Study A case study is one of the common ways to demonstrate the effectiveness or suitability of explanations obtained from explainable recommender systems. It typically involves choosing at least one user and showing his/her recommendation obtained from the model along with its explanation. By comparing them with the user's personal profile or other evidence of the user's interests, it is possible to evaluate whether this explanation makes sense or is suitable for being used in real-world situations or not. For example, in [90], a sampled user along with his/her recommendations and their explanations were shown in order to investigate how the model captures this user's semantic attribute preferences. In [129], some entities in a HIN with their important scores towards the selected users were presented. In [96], apart from using heat maps, the authors also showed the most important neighbors and the most important multi-style meta-paths of the selected users. In [98], the proposed model KGIN learned intents that represent users' preferences by attentively aggregating information from multiple relations. To demonstrate the explainability of this model, examples of these intents were presented. For each intent, the relations in a HIN that formed this intent along with their weights were extracted and shown as a case study. This allowed the authors to identify which relations were important for a given user and gain some knowledge of how KGIN modeled the users' preferences.

As for those models that produce path-level explanations, the explanation paths of randomly sampled users can be illustrated for evaluation. For instance, in [91], the authors randomly selected a user and generated his/her top-3 recommendations along with their explanation paths. These explanations were presented and their suitability was discussed. Similarly, for PLM-Rec [103], case studies based on the randomly selected users were conducted for evaluation. These case studies allowed the authors to demonstrate the performance of

PLM-Rec in terms of solving recall bias. As for RippleNet [61], the explanation paths of a randomly sampled user were compared among its variations. The relevance probabilities of these paths were also discussed. The decentralization of these probabilities was shown in their results. For CAFE [95], two case studies of two chosen users were shown in the paper. For each user, his/her layout tree obtained from his/her profile and the subset of generated explanation paths based on his/her layout tree were presented. In HAGERec [62], KGAT [5] and KGAT+ [102], paths connecting between the randomly chosen user-item pair were extracted with the help of attention mechanisms. These explanation paths revealed the user's preferences from different perspectives and were examined for evaluation. For TMER [73], the authors randomly selected a user and retrieved his/her previously interacted items. Then item-item path instances connecting each pair of items were extracted based on the attention weights learned by their proposed model. These paths formed the explanation paths of this user and were analyzed based on their suitability and practicality. For those models using RL-based path reasoning methods to provide explanations, i.e., PGPR [65], MKRLN [101], TPRec [104] and ReMR [68], explanation paths discovered via path navigation using the RL agents were extracted and presented to exhibit the performance of their explanation generating methods. Apart from presenting explanation paths individually for evaluation, there are some papers that combined such paths to form a sub-graph and used it to examine the explainability of their models. For example, in [70] and [64], the authors extracted all the explanation paths connecting the selected user-item pair and presented them in the form of a sub-graph showing different high-order connections between them.

For those models that derive rules as explanations such as RuleRec, a case study was also conducted in order to give some examples of rules and how they can be applied. As for RuleRec [69], two positive rules learned from this model were first derived to examine the explainability. These rules indicated that if there is a path connecting a user's unobserved item and one of the user's items, then this unobserved item is more likely to be recommended. To verify the practicality of these rules, they first examine whether item pairs connected with these rules correspond with common sense or not. Then, these rules were labeled by three experts to check whether users would agree with these rules.

Apart from the practicality of explanations, case studies were conducted to evaluate the fairness/diversity of explanation paths as well. The explanation paths obtained from the fairness-aware algorithm FairKG4Rec [94] were extracted and compared with other baselines in terms of path diversity. The variety of nodes and relations within these extracted paths were examined in order to evaluate their diversity. This exemplifies the uses of case studies to evaluate the explainability of HIN-based recommender systems in different aspects not only accuracy or practicality. Similarly to visualization, case studies are also subjective. They require human interpretation which can be highly varied depending on prior knowledge and experience. Thus, using case studies may not be sufficient to explicitly evaluate the performance of explanations generated by the models.

User Survey Conducting a user survey is another straightforward way to evaluate explainability since it can provide users' opinions towards generated explanations or explainable recommendations. This evaluation approach involves recruiting a group of people (which could be stakeholders or potential users of the systems) and using questionnaires/tests to gain knowledge from these users regarding the explainability of recommender systems. For instance, in [86], the persuasiveness of explanations generated by SemAuto was evaluated through online A/B testing. In the first step, the user had to select at least 15 movies that they have watched from the list provided and rate each movie on a five-star rating scale. These movies were treated as input data for their model. After that, the user was given a list of recommendations based on their selected movies without any explanations. The user then was asked to rate the recommended items. Next, the explanations were presented to the user and the user was asked to re-rate the top-2 recommended items. The results before and after providing the explanations were then compared to examine the impact of the explanation in terms of persuasiveness.

In [90], a user study was conducted to answer whether the number of semantic attributes in the explanation impacts user satisfaction or not. The authors assumed that user satisfaction can be achieved by recommending an item with an explanation that contains a higher number of semantic attributes. With this hypothesis, 34 participants were assigned to either the low, medium, or high group where the number of semantic attributes used in the explanations is up to one, up to three, and up to five, respectively. Some demographic information of each participant was collected including age, gender, major of study, weekly hours watching movies, and his/her favorite movie semantic attributes. Each participant was asked to rate, on a 1-to-5 scale, at least 10 movies they have previously watched from a selection of movies. Then, a recommendation with an explanation, containing different numbers of attributes based on the participant's assigned group, was provided to the participant. The participant was asked to fill out a Likert Scale questionnaire regarding the explanation provided, for example, "This explanation helps me understand why this movie was recommended" and "Based on the share of semantic attributes between the recommended movie and my interest in these semantic attributes, I will watch this movie". The results of this survey were presented along with analytical testing including an Analysis of Variance (ANOVA) and a Tukey's Honestly Significant Difference (HSD) to determine the significance of the results.

In [100], two human evaluations were conducted to answer two questions: (1) whether the generated explanation paths from UCPR are relevant to the corresponding highlighted entities and (2) whether the highlighted entities in the user's portfolio are sufficient for anticipating the positive item of the user. For the first question, the participants were asked to rate the relevance between the explanation paths and the highlighted nodes learned from the model at each step. The rating criteria were proposed to determine the relevance. For the second question, the participants had to choose the next nodes based on the highlighted entities and the previously chosen nodes until they reached the final item node. In [99], the authors conducted a user survey to evaluate the explanation paths. The authors sampled 50 paths connecting a user to one of his/her previously interacted items in the training set to represent examples of users' historical behaviors. The participants were asked to rank three models based on the consistency between their generated explanation paths and the users' historical behaviors. In [135], 100 user-item pairs were randomly chosen and the explanation paths for these pairs were generated. Then, 10 participants were asked to evaluate the Relevance and Diversity scores of these explanation paths. Relevance indicates how likely the explanations are related to the recommended items. Diversity shows whether the explanations consist of various types of nodes and relations or not. Also, to avoid inconsistency in the results from different participants, the explanation paths were divided into 5 groups in which each group contains 100 paths corresponding to certain 20 user-item pairs. For each group, two participants were assigned and the final scores were obtained from the average scores among these two participants.

To evaluate the counterfactual explanations generated by PRINCE [93], three recommendation items were shown to 500 participants on Amazon Mechanical Turk (AMT). For each recommendation, two explanations, i.e., a counterfactual explanation and an explanation path connecting the user to the item were presented. The participants were requested to answer three questions, "Which method do you find more useful?", "How do you feel about being exposed through explanations to others?" and "Personally, which type of explanation matters to you more". This is to examine whether the counterfactual explanations were more useful

Table 2.4: Quantitative methods adopted in state-of-the-art explainable HIN-based recommender systems

Quantitative method	Models
Mean Explainability Precision (MEP)	SEP [87], LDSDMF [90]
Mean Explainability Recall (MER)	SEP [87], LDSDMF [90]
Mean Explainability F-Score (xF-SCORE)	LDSDMF [90]
Performance shift	SEP [87], Liu et al.'s [129]
Simpson's Index of Diversity (SID)	FairKG4Rec [94], UCPR [100]
Review matching	ADAC [67], UCPR [100]
Jensen–Shannon (JS) divergence	LOGER [99]
Levenshtein distance	KR-GCN [135]

and preferred than the path-based explanations given the same recommendations. The participants were also asked to rate different explanations on a scale from 1 to 3 being "Not useful at all", "Partially useful", and "Completely useful" respectively. Despite the valuable insights into human behaviors, conducting a user survey typically requires a lot of human experts and numerous afford and resources. This is a challenge of using a user survey to evaluate the explainability of recommender systems.

2.3.3.2 Quantitative Evaluation

Quantitative methods are evaluation methods that involve using quantitative metrics to measure and compare the explainability or suitability of the generated explanations or explainable recommendations. Compared to qualitative methods, quantitative methods are more concrete and definite since they are not varied by human deliberation. In this section, we discuss the quantitative evaluation metrics that have been proposed in state-of-the-art explainable HIN-based recommender systems. They are summarized in Table 2.4. More details of these metrics are described in the following paragraphs.

Mean Explainability Precision (MEP), Mean Explainability Recall (MER) and Mean Explainability F-Score (xF-SCORE) In [87], the applicability of the models in terms of predicting explainable items was evaluated by two metrics, i.e., Mean Explainability Precision (MEP) and Mean Explainability Recall (MER) [166, 167]. MEP and MER were defined similarly to regular Precision and Recall to measure the accuracy of predicting those items that are explainable to the users. Compared to regular Precision and Recall, instead of considering a set of recommended items and a set of positive items for each user, MEP and MER consider a set of recommended items and a set of explainable items for each user. To obtain a set of explainable items for each user, different strategies can be used. For instance, it can be a set of items with at least one explanation generated by a post-hoc explaining method [87] or a set of items whose explainability values are higher than a pre-defined threshold [72]. In [90], in addition to MEP and MER, Mean Explainability F-Score (xF-SCORE) was also used. This xF-SCORE can be computed by MEP and MER similar to the regular F-SCORE computed by Mean Average Precision (MAP) and Mean Average Recall (MAR).

Performance shift In [87], the shift in their model performance was considered to examine whether the nodes and relations in the generated explanation paths are significant or not. Specifically, given a user-item pair and its explanation path, the nodes and relations in this path were removed from the training data. Then, the recommendation model was re-trained based on this altered training data. This re-trained model was used to evaluate the explanation path based on the assumption that if the rating score of this user-item pair decreases significantly after re-training, then the removed nodes and relations are significant for this score prediction. In that case, explanation paths contain significant information that reflects how the recommendation model works. In [129], the generated explanation in the forms of significant concepts (or attributes) was evaluated by adversarial perturbation [168] similarly to [87]. The idea is that an attribute can serve as a good explanation if the removal of this attribute notably changes the model prediction. In other words, it has a high influence on the model prediction. For evaluation, 200 users were first sampled from the test set. The recommendations for these users were predicted and the important item nodes or other attribute nodes were identified for each pair of a user and his/her recommended item. After removing these important item/attribute nodes, the new recommendations were generated and compared with the original recommendations. The Recall shift (the difference in Recall values) between the new and the original recommendations was computed. A higher value of this shift means that the explanations are accurate and can highly reflect users' preferences.

Simpson's Index of Diversity (SID) In [94] the group-level unfairness of the explanation path diversity was defined based on Simpson's Index of Diversity (SID). It indicates the diversity of a given user's historical interactions with explanation paths. Ranging from 0 to 1, the higher the SID is, the more diverse the explanation paths. Based on SID, FairKG4Rec was compared with the other baselines in terms of diversity and fairness in recommendations and explanation paths. SID was also adopted in [100] to quantify the diversity of explanation paths obtained from UCPR.

Review matching In [67], the ground-truth reviews were used to evaluate the explainability of the explanation paths. They assumed that the explainability of an explanation path can be achieved if it contains several nodes that appear in the ground-truth review. First, the review words were filtered by considering their frequencies and TF-IDF values. The remaining words in the reviews were treated as ground-truth reviews. Then, the explanation paths generated by their model ADAC were matched with these ground-truth reviews. After matching, Precision and Recall based on the top-5 matched nodes were used for comparison. Similarly, UCPR [100] also evaluated the explanation paths discovered by the RL agent by conducting a review matching. The Recall and NDCG metrics were applied based on the top-10 matched entities to justify the effectiveness.

Jensen–Shannon (JS) divergence In [99], apart from the user study, the authors also conducted a quantitative evaluation of the faithfulness of the explanation paths. Inspired by [169, 170, 171], the Jensen–Shannon (JS) divergence of rule-related distributions from training and test sets were adopted to measure this quality. First, 50 users from the training set were randomly selected. For each user, around $1,000$ paths connecting this user to his/her item were sampled. Based on these paths, the JS score was computed. For the test set, the explanation paths connecting each user in the test set and his/her item were first generated. Then, the JS score over these paths was calculated. Faithfulness was assessed by considering these two scores. The smaller the values of these two JS scores, the better the faithfulness of the generated explanation paths.

Levenshtein distance In [135], the sub-graphs containing the explanation paths were used to examine the explainability performance. Given a user-item pair, the top-3 explanation paths were extracted. These explanation paths together formed a sub-graph showing various reasoning paths from a given user to his/her recommended item. Similar sub-graphs generated from their proposed model KR-GCN and the other baselines were compared in terms of their topological similarity by using Levenshtein distance [172]. This indicated reasoning sub-graphs obtained from KR-GCN have some overlap with those obtained from the baselines. In other words, their model KR-GCN was capable of generating reasoning sub-graphs as the baselines.

Since most visually-aware recommender systems rely on extracted latent visual features, they often lack transparency and explainability. This deficiency can give rise to a lack of confidence and doubt among users regarding these systems. Therefore, the task of improving the explainability of visually-aware recommender systems is undoubtedly necessary, yet remains challenging [16]. As evident in various explainable HIN-based recommender systems, HINs are valuable resources highly suitable for providing explainability in recommendations. Developing HIN-based recommender systems capable of incorporating visual information from images offers a solution for generating explainable visually-aware recommendations. However, as mentioned earlier, existing HIN-based recommender systems often overlook visual information. This gap highlights the need to bridge visual information integration into HINs to develop explainable visually-aware recommender systems based on HINs.

2.4 Summary

Previous studies have exhibited the possibilities of using side information apart from user-item interactions to build hybrid recommender systems [12]. These recommender systems can leverage both collaborative filtering and content-based filtering techniques to enhance their performances. One common type of side information that has been used is visual information from item images [16]. Visual information can be used to discover users' visual preferences, which is highly advantageous, especially in certain domains such as clothing items. Due to this benefit, many visually-aware recommender systems that consider visual information were proposed. Since most visually-aware recommender systems leverage latent image features, they typically lack transparency and explainability [16]. It can lead to distrust and skepticism from users towards these systems. Therefore, improving the explainability of visually-aware recommender systems is undoubtedly necessary and still challenging.

HIN-based recommender systems offer an additional advantage beyond accuracy: the ability to provide explanations for recommendations. Unlike conventional and deep-learning recommender systems that often lack transparency, HIN-based systems can offer clear reasoning behind each recommendation. They achieve this by utilizing the structure and connections within HINs, ensuring recommendations are interpretable. This transparency not only builds user trust but also allows for personalized recommendations based on user preferences. Although existing HIN-based recommender systems have succeeded in accuracy and explainability, they mainly focus on semantic information such as user-item metadata within HINs. They typically disregard visual information, creating a gap in leveraging it. By addressing this gap, HINs could efficiently enable visually-aware recommendations while maintaining explainability.

The following chapters will focus on bridging this gap by proposing a novel approach that integrates visual information into HINs and a novel explainable visually-aware recommender system framework. This approach aims to leverage the strengths of both visual information and HINs, enhancing the accuracy and explainability of visually-aware recommendations. Through a comprehensive exploration of this approach, this thesis aims to contribute to the advancement of transparent and accurate recommendations, thereby enriching the user experience and confidence in the recommendations provided by these systems.

Chapter 3

Visually-Augmented Heterogeneous Information Networks and Visually-Aware Recommendations

As mentioned in Chapter 1, HINs have been ubiquitous in recommendation research due to their capability of providing contextual information that can overcome the sparsity problem as well as explainability in recommendations [5, 64, 148, 174, 175]. However, most HIN-based recommender systems mainly focus on semantic information and ignore visual information. Therefore, in accordance with the first research question in Section 1.2, this chapter introduces a novel method that integrates visual features into a HIN, allowing the simultaneous utilization of visual information and multi-hop relations for visually-aware recommendations.

In this chapter, a method for constructing a new type of HINs called *visually-augmented HINs* is presented. These augmented HINs incorporate both semantic information from user and item metadata and visual information from visual factor nodes and visual relations. These nodes and relations are generated based on image features extracted using various image feature extraction methods. Visually-augmented HINs are valuable for learning latent representations of users and items, profiling users' preferences from both semantic and visual perspectives. However, it is noteworthy that the majority of existing approaches [85, 133] only consider semantic information. To address this limitation, an approach for learning user latent representations from a hybrid context that encompasses both semantic and visual infor-

This chapter has been published in [173] available at `https://doi.org/10.1145/3397481.3450686` mation is proposed in this chapter. To create this hybrid context, a novel type of meta-path called *probabilistic meta-paths* is introduced. These meta-paths are utilized to create a hybrid context for learning representations that can be employed in recommender systems. Finally, this chapter outlines a visually-aware recommendation approach based on visually-augmented HINs and the proposed user latent representation learning method. This integrated approach holds the potential to significantly improve the accuracy and interpretability of visually-aware recommendations. The contributions in this chapter are summarized as follows:

- A novel method is proposed for constructing visually-augmented HINs containing both semantic information from user and item metadata and visual information in the forms of visual factor nodes and visual relations.
- A new type of meta-path called probabilistic meta-paths is introduced to create a hybrid context for learning representations suitable for visually-aware recommender systems.
- A method for learning user latent representations from the hybrid context of semantic and visual information is proposed.
- A visually-aware recommendation approach based on visually-augmented HINs and utilizing probabilistic meta-paths is proposed.

The rest of this chapter is organized as follows: Section 3.1 provides preliminaries as a foundation for comprehending the proposed visually-augmented HINs. Section 3.2 introduces the definition of visually-augmented HIN and the proposed method to construct a visually-augmented HIN. Section 3.3 discusses probabilistic meta-paths used for forming users' hybrid contexts. Section 3.4 explains an approach used for learning user latent representations from a hybrid context. Also, in this section, a visually-aware recommendation model using these representations is discussed. Section 3.5 provides details on the experiments conducted to evaluate the performance of this model as well as the proposed visually-augmented HINs and probabilistic meta-paths. Finally, the experimental results and discussions are provided in Section 3.6.

3.1 Preliminaries

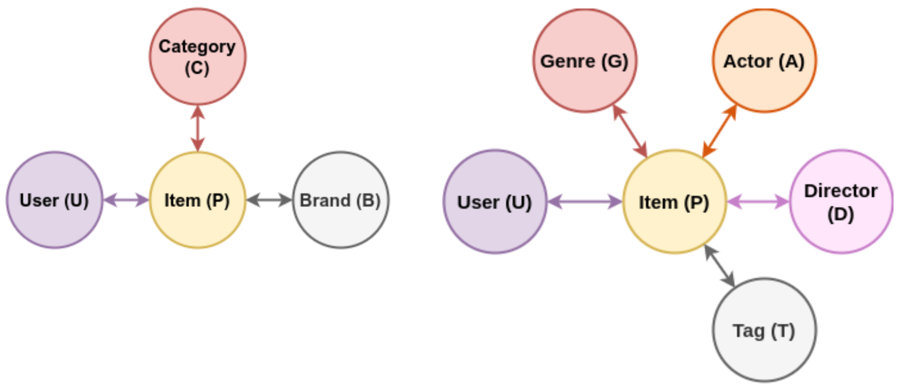
As previously mentioned, constructing a visually-augmented HIN involves integrating visual information (i.e., image features) extracted from item images into a regular HIN. To establish a comprehensive foundation for discussing the proposed augmentation method, this section initially introduces the definitions of HIN and meta-path, explains a meta-path based node embedding method called metapath2vec, and provides detailed explanations of some existing image feature extraction methods that can be adopted for constructing visually-augmented HINs.

3.1.1 Heterogeneous Information Network

A HIN is a type of network that models diverse types of entities and their relationships in a network/graph. In other words, a HIN is a network that incorporates different types of nodes and relations, representing various objects and their attributes. For example, in a movie domain, nodes can represent actors, directors, movies, and genres, while edges can represent relationships such as "acted in," "directed," and "belong to". HINs are becoming increasingly important in various fields, including recommendation systems, due to their ability to represent complex and diverse relationships among entities. They have been shown to be effective in improving recommendation systems. By utilizing relationships among different entities, the recommendations can be more personalized and accurate. HIN schema and HIN can be defined as follows:

Definition 3.1. (HIN schema) *[63] Let* $\mathbb{G} = (\mathbb{N}, \mathbb{R}, \mathbb{W})$ *denote a HIN schema consists of a set of node types* $\mathbb{N}$*, a set of relation types* $\mathbb{R}$ *and a non-negative weight function* $\mathbb{W} : \mathbb{R} \to \mathfrak{R}$ *that maps each relation type to a non-negative real value 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* N_i *to* 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*.*

Example 3.1. (HIN Schema) Figure 3.1 shows an example of a HIN schema in two different domains: clothing and movie domains. In this example, there are four node types in the clothing domain schema: user node type (U), item node type (P), category node type (C), and brand node type (B). Among these node types, there are six relation types: user-item relation type (R_{UP}) and its inverse (R^{-1}_{UP}), item-category relation type (R_{PC}) and its inverse (R^{-1}_{PC}), item-brand relation type (R_{PB}) and its inverse (R^{-1}_{PB}). Note that R^{-1}_{UP}, R^{-1}_{CP} and R^{-1}_{PB} are equivalent to item-user relation type R_{PU}, item-category relation type R_{PC} and item-brand relation type R_{BP} respectively. In the movie domain schema, there are six node types: user node type (U), item node type (P), genre node type (G), actor node type (A), director node type (D), and tag node type (T). Similar to the clothing domain schema, the



(a) Clothing (b) Movie

Figure 3.1: Example of HIN schemas in two item domains: (a) clothing items and (b) movie items. These schemas illustrate all node and relation types within the HIN, with circles representing node types and arrows denoting relation types between the connected node pairs.

movie domain schema consists of ten relation types: R_{UP}, R_{UP}^{-1}, R_{PG}, R_{PG}^{-1}, R_{PA}, R_{PA}^{-1}, R_{PD}, R_{PD}^{-1}, R_{PT}, and R_{PT}^{-1}.

Definition 3.2. (HIN) *[63] Given a HIN schema $\mathbb{G} = (\mathbb{N}, \mathbb{R}, \mathbb{W})$, a HIN is defined as a weighted and 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: $\phi : \mathcal{N} \to \mathbb{N}$ and $\psi : \mathcal{R} \to \mathbb{R}$ respectively. Given nodes $x, y \in \mathcal{N}$, $r_{x,y}$ denotes a relation from x to y and its weight is denoted by $w(x, y) = \mathbb{W}(\psi(r_{x,y}))$.*

3.1.2 Meta-Path

In a HIN, a meta-path is a high-level abstraction of the structural relationship between different types of nodes in the network. It is defined as a sequence of node types and the corresponding relation types that connect them. In other words, a meta-path is a pattern that specifies a type of relationship between different types of nodes in a HIN. For example, a meta-path in a movie database could be actor-director-movie, which represents the sequence of nodes (actor, director, movie) and the relations (acted-in, directed) that connect them. Mathematically, a meta-path can be defined as follows:

Definition 3.3. (Meta-Path) *[71] Given a HIN $\mathcal{G}$, a meta-path m is defined as*

$$N_1 \xrightarrow{R_{N_1,N_2}} N_2 \cdots N_l \xrightarrow{R_{N_l,N_{l+1}}} N_{l+1} \tag{3.1}$$

(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 N_1 and N_{l+1} where $\circ$ denotes the composition operator on relations. l is the length of m, $\mathbb{N}_m = \{N_1, N_2, ..., N_{l+1}\}$ is the set of node types in m and $\mathbb{R}_m = \{R_{N_1,N_2}, \cdots, R_{N_l,N_{l+1}}\}$ is the set of relation types in m. A path $z = (n_1n_2 \cdots n_{l+1})$ in $\mathcal{G}$ is called a ***path instance*** *of m, if each n_i belongs to type N_i in m for all $i = 1, 2, ..., l + 1$.*

Meta-paths are used to capture the similarity between different nodes in the HIN, and they can be used for a variety of tasks, such as entity recommendation, clustering, and link prediction. They provide a way to focus on specific types of relationships in the network and can be used to discover meaningful patterns and relationships that are not readily apparent from the raw data. In recommender systems, meta-paths can be used to identify related items that are not directly connected in a HIN. For example, if a user has expressed interest in a particular actor, the system can use a meta-path that includes the actor, director, and movie nodes to recommend other movies that the user may be interested in. This allows meta-paths to be a powerful tool for analyzing and understanding complex relationships in HINs to model users' personal preferences. Due to this unique advantage of meta-paths, numerous methods for node embedding based on meta-paths have been proposed [176]. The following section discusses one of the ubiquitous meta-path based node embedding methods.

3.1.3 Metapath2vec

Metapath2vec is a node embedding method that is designed to learn representations of nodes in HINs using meta-paths. Unlike homogeneous networks, which contain only one type of node and one type of edge, HINs consist of multiple types of nodes and relations. Metapath2vec is particularly effective for learning embeddings of nodes in such networks because it takes into account the different types of nodes and relations and their relationships. To accomplish this, metapath2vec utilizes the concept of meta-paths, which are sequences of node types and edge types that describe the desired semantic relationship between nodes in the network.

The metapath2vec algorithm is based on the skip-gram model, which is a popular method for learning word embeddings in natural language processing. The objective of the skip-gram model is to predict the context words given a target word, or vice versa. To learn node embeddings in metapath2vec, the same idea is applied to predict the context nodes given a target node. Specifically, given a HIN $\mathcal{G} = (\mathcal{N}, \mathcal{R}, \mathcal{W})$ with a schema $(\mathbb{N}, \mathbb{R}, \mathbb{W})$, for each $N_i \in \mathbb{N}$, let $\mathcal{N}_i$ be the set of nodes in $\mathcal{N}$ with the type N_i. For each node $n \in \mathcal{N}$, let $\mathcal{N}_i^n$ denote the set of neighbor nodes with the type N_i of n. Node embeddings are computed by maximizing the probability of having the neighborhood context $\mathcal{N}_i^n$, $N_i \in \mathbb{N}$, given a node $n \in \mathcal{N}$ as follows:

$$\underset{\Theta_m}{\operatorname{argmax}} \sum_{n \in \mathcal{N}} \sum_{N_i \in \mathbb{N}} \sum_{n_i \in \mathcal{N}_i^n} \log p(n_i | n; \Theta_m) \tag{3.2}$$

where $p(n_i|n;\Theta_m)$ is a softmax function defined as:

$$p(n_i|n;\Theta_m) = \frac{\exp(e_{n_i} \cdot e_n)}{\sum_{n' \in \mathcal{N}} \exp(e_{n'} \cdot e_n)} \tag{3.3}$$

where Θ_m represents the parameters and e_{n_i}, e_n and $e_{n'}$ are the embedding vectors of node n_i, node n and node n' respectively. To efficiently optimize this probability, the negative sampling strategy proposed in [177] is adopted. This strategy uses a relatively small set of nodes sampled from the HIN instead of the set of all nodes in the HIN. Given a negative sample size K_m, the objective of metapath2vec is finally defined as follows:

$$O(\Theta_m) = \log \sigma(e_{n_i} \cdot e_n) + \sum_{k=1}^{K_m} \mathbb{E}_{n^k \sim X}[\log \sigma(-e_{n^k} \cdot e_n)] \tag{3.4}$$

where

$$\sigma(x) = \frac{1}{1 + \exp(-x)} \tag{3.5}$$

and X is the pre-defined distribution from which a negative node n^k is drawn for K_m times.

To determine the neighborhood context of node n, $\mathcal{N}_i^n$, meta-path based random walks are performed. Given a meta-path m, $\mathcal{N}_i^n$ is defined as a set of nodes that can be reached by meta-path based random walks based on m starting from node n. Figure 3.5 shows an example of how the neighborhood context of the "User 1" node is formed based on meta-path random walks with meta-path $UPCP$. By following this meta-path, two item nodes, "T-shirt C" and "T-shirt B", can be put into the same context as "User 1" since they can be reached by the meta-path based random walks starting from "User 1", i.e., ("User 1", "T-shirt A", "Category: T-shirt", "T-shirt C") and ("User 1", "T-shirt A", "Category: T-shirt", "T-shirt B").

From Eq. 3.3 and 3.4, metapath2vec considers different types of nodes homogeneously and draws negative sample nodes regardless of their types. To consider node type information, the improved version of metapath2vec called *metapath2vec++* was also proposed in [85]. In metapath2vec++, the softmax function in Eq. 3.3 is normalized with respect to the node type of the context n_i, i.e.,

$$p(n_i|n;\Theta_m) = \frac{\exp(e_{n_i} \cdot e_n)}{\sum_{n_j \in \mathcal{N}_j} \exp(e_{n_j} \cdot e_n)} \tag{3.6}$$

where $\mathcal{N}_j$ denotes the set of nodes with the type N_j. The sampling distribution is also specified by the node type of the neighborhood context n_i. Thus, from Eq. 3.4, the objective of metapath2vec++ is defined as

$$O(\Theta_m) = \log \sigma(e_{n_i} \cdot e_n) + \sum_{k=1}^{K_m} \mathbb{E}_{n_j^k \sim X_j} [\log \sigma(-e_{n_j^k} \cdot e_n)], \tag{3.7}$$

where X_j is the pre-defined distribution from which a negative node n_j^k with the type N_j is drawn for K_m times. The parameters e_{n_i}, e_n, and $e_{n_j^k}$ are updated by using the stochastic gradient descent algorithm whose gradients are computed as follows:

$$\frac{\partial}{\partial e_{n_j^k}} O(\Theta_m) = (\sigma(e_{n_j^k} \cdot e_n - \mathbb{I}_{n_i}[n_j^k]))e_n \tag{3.8}$$

$$\frac{\partial}{\partial e_n} O(\Theta_m) = \sum_{k=0}^{K_m} (\sigma(e_{n_j^k} \cdot e_n - \mathbb{I}_{n_i}[n_j^k]))e_{n_j^k} \tag{3.9}$$

where $\mathbb{I}_{n_i}[n_j^k]$ is an indicator function indicating whether n_j^k is the neighborhood context node n_i and when $k = 0$, $n_j^0 = n_i$.

3.1.4 Image Feature Extraction Methods

In this subsection, several popular image (visual) feature extraction methods are discussed. This thesis considers five methods: SIFT (Scale-Invariant Feature Transform)[25], SURF (Speeded Up Robust Features)[26], ORB (Oriented FAST and Rotated BRIEF)[27], Color Histogram[178], and latent feature extraction using a pre-trained CNN model [179]. Widely applied in various computer vision tasks such as image classification [180, 181], image retrieval [179, 182, 183], and object recognition [184, 185], these methods are chosen in this thesis for extracting visual features to understand users' visual preferences. The following paragraphs describe each method, covering their algorithms, advantages, and disadvantages.

SIFT (Scale-Invariant Feature Transform) SIFT [25] is an image processing technique used for feature detection and extraction in gray-scale images. It has been widely used in various computer vision applications. SIFT features are detected by identifying extrema in the Difference of Gaussians (DoG) scale space, and then their descriptors are generated by calculating gradient orientations and magnitudes in local regions around these keypoints. Histograms of these gradient orientations are then combined to create a descriptor vector that encapsulates the characteristics of keypoints. SIFT is invariant to scale and rotation. This means that it can detect the same feature no matter whether the object is scaled or rotated.

Also, SIFT is robust to changes in illumination. It can detect features in different lighting conditions. However, SIFT is sensitive to noise in the image, which can lead to false detections or inaccurate feature matching. Using it can also be computationally expensive and requires a significant amount of memory to store the extracted features.

SURF (Speeded Up Robust Features) SURF [26] is an image processing technique that is used to detect and describe keypoints in gray-scale images similar to SIFT. The key idea of SURF is to efficiently compute keypoint features by using a series of approximations that are less computationally expensive than other similar techniques. The SURF algorithm is based on the same principles and steps as SIFT with slight differences. As opposed to SIFT, SURF uses an approximation method that allows for faster and more robust computation, at the cost of slightly lower accuracy in some cases. Additionally, SURF uses a different descriptor computation method that is based on the responses of Haar wavelet filters, while SIFT uses a Gaussian-weighted histogram of gradients. However, one disadvantage is that SURF is not as invariant to affine transformations as SIFT. It may not perform well in highly textured or cluttered images.

ORB (Oriented FAST and Rotated BRIEF) ORB [27] is another feature extraction method applied to gray-scale images. It was developed as an alternative to SIFT and SURF feature extraction methods in computation cost, matching performance, and license fees since SIFT and SURF are patented. Therefore, ORB is widely adopted in both academic and commercial areas. ORB is based on the FAST (Features from Accelerated Segment Test) [186] keypoint detector and a modified version of the visual descriptor BRIEF (Binary Robust Independent Elementary Features) [187]. The ORB method is computationally less expensive than the SIFT method because it only requires one Gaussian filter per scale level, while SIFT requires multiple filters per scale level. Additionally, ORB uses a faster feature descriptor compared to SIFT, making it suitable for real-time applications. However, SIFT and SURF are generally more robust and accurate than ORB because of their more complex scale space construction and feature descriptors.

Color Histogram A color histogram is a histogram of the distribution of colors in an image, indicating the number of pixels with a specified color in each of a fixed list of color ranges. This list of colors depends on the color space of the images. For example, in an RGB color space, there are three colors in the list: red, green, and blue. A color histogram ignores spatial locations of colors and instead focuses solely on the proportion of different colors present in an image. It is ideal for recognizing objects, distinguishable by colors, even if their position and rotation are unknown. In this work, the HSV (Hue-Saturation-Value) color space is considered for generating color histograms of item images. The HSV color space is designed to be more perceptually uniform than other color spaces such as RGB. Changes in the values of hue, saturation, and value are more closely related to human perception of color than other color spaces. Therefore, it is easier to understand, manipulate, and leverage color information in this color space. Another significant advantage of the HSV color space is that it separates color information from brightness information. It can be helpful in particular image processing tasks where accurate processing of color information is required such as color thresholding or object recognition. In this work, to effectively capture color information, both hue and saturation dimensions are considered when generating color histograms. The range of hue is $[0, 179]$ and the range of saturation is $[0, 255]$. The size of each bin is 1; thus the total number of bins is $46,080$ which is the total number of possible hue degrees (180) multiplied by the total number of possible saturation values (256). Therefore, a color histogram generated from each item image is $46,080$-dimensional.

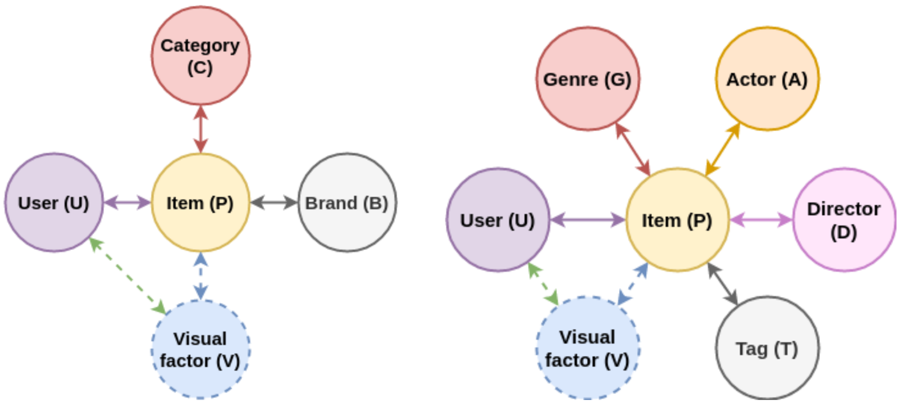
Latent features Convolutional Neural Networks (CNNs) are a type of deep neural network that has revolutionized the field of computer vision. These networks can automatically learn a hierarchy of visual features from raw pixel data. Pre-trained CNN models are typically trained on large datasets such as ImageNet [149] to extract high-level image features. These pre-trained models can be used for feature extraction for many downstream computer vision tasks such as image recognition and classification tasks. There are several ways to extract image features from pre-trained CNN models. One common approach is to use the output of the last convolutional layer of the model as the features. This layer extracts the most abstract and high-level features of an image. These features can then be flattened and fed into a classifier or other downstream models. One commonly used pre-trained CNN model is *CaffeNet* [136]. CaffeNet is a CNN implemented in the Caffe (Convolutional Architecture for Fast Feature Embedding) deep learning framework. Its architecture is based on the AlexNet [188] architecture. CaffeNet consists of 8 layers, including 5 convolutional layers and 3 fully-connected layers. It was trained on the ImageNet dataset and has been extensively used to extract visual features for many downstream tasks including recommendation [37]. Typically, the output of the second fully-connected layer, namely "FC7", is extracted to obtain a 4096-dimensional visual feature vector for each image.

3.2 Construction of Visually-Augmented Heterogeneous Information Network

A HIN typically contains only semantic factors which are attributes, characteristics, or features of users and items that are derived from user and item metadata. Each individual factor has a different influence on the recommendation made for users. Some examples of semantic factors are category and brand for a clothing item domain and genre, actor, and director for a movie domain. Multi-hop relations extracted from such a HIN can represent only users' preferences based on these semantic factors. However, one of the key features of HINs is that various types of information, such as text, images, and other metadata, can be integrated into a single network. This enables the representation of complex relationships among different entities. Therefore, in this work, images are integrated into HINs in order to facilitate visually-aware recommendation learning. Specifically, this work proposes a *visually-augmented HIN* that contains pivotal visual factors from item images defined as follows:

Definition 3.4. (Visually-augmented HIN schema) *Given a HIN schema $\mathbb{G}$, a visually-augmented HIN schema is defined as $\mathbb{G}' = (\mathbb{N}', \mathbb{R}', \mathbb{W}')$ where $\mathbb{N}' = \mathbb{N} \cup \{V\}$, $\mathbb{R}' = \mathbb{R} \cup \{R_{UV}, R_{PV}\}$, $\mathbb{W}' : \mathbb{R}' \to \mathfrak{R}$ where V is a visual factor node type and R_{UV} and R_{PV} are visual relation types connecting a user node to a visual factor node and an item node to a visual factor node respectively.*

Example 3.2. (Visually-Augmented HIN Schema) Figure 3.2 shows an example of a visually-augmented HIN schema in a clothing domain and a movie domain. In this figure, all nodes and edges shown in solid lines are all semantic types. In the clothing domain schema, there are four semantic node types: user node type (U), item node type (P), category node type (C), and brand node type (B). In the movie domain schema, there are six semantic node types: user node type (U), item node type (P), genre node type (G), actor node type (A), director node type (D), and tag node type (T). They belong to the original HIN schema shown in Figure 3.1. A visual factor node type (V) is added to the original schema along with two new relation types, i.e., user-visual factor relation (R_{UV} and R_{UV}^{-1}) and item-visual factor relation (R_{PV} and R_{PV}^{-1}). These additional nodes and edges are presented with dash lines in this figure.



(a) Clothing (b) Movie

Figure 3.2: Example of visually-augmented HIN schemas in two item domains: (a) clothing items and (b) movie items. The semantic node and relation types in the original HIN are depicted with circles and arrows featuring solid line borders, while the additional visual factor node type (blue circles) and visual relation types (green and blue arrows) are depicted with circles and arrows featuring dashed borders.)

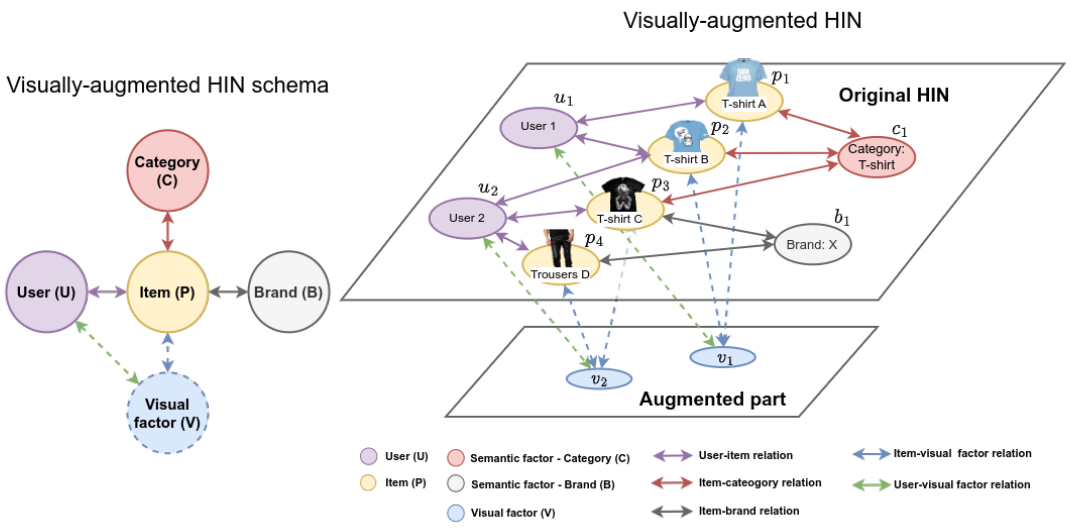


Figure 3.3: Example of visually-augmented HIN schema and visually-augmented HIN. In the visually-augmented HIN, the upper plane represents the original HIN consisting solely of semantic factor nodes, while the lower plane depicts the augmented part of the visually-augmented HIN containing visual factor nodes.

Definition 3.5. (Visually-Augmented HIN) *Let $\mathcal{V}$ denote a set of visual factor nodes and $\mathcal{R}_V$ denote a set of relations connecting semantic and visual factor nodes. A visually-augmented HIN $\mathcal{G}' = (\mathcal{N}', \mathcal{R}', \mathcal{W}')$ is a HIN with a schema $\mathbb{G}'$ where $\mathcal{N}' = \mathcal{N} \cup \mathcal{V}$, $\mathcal{R}' = \mathcal{R} \cup \mathcal{R}_V$ and $\mathbb{W}' : \mathbb{R}' \rightarrow \mathfrak{R}$ denotes a non-negative weight function that maps each relation type to a real value in $\mathfrak{R}$.*

Example 3.3. (Visually-Augmented HIN) Figure 3.3 shows a visually-augmented HIN with the schema defined in Example 3.2. The above plane presents a regular HIN with only semantic factors, i.e., category and brand. On the below plane, there are two visual factor nodes connected to user and item nodes via user-visual factor relations and item-visual factor relations.

Figure 3.4 illustrates the overall framework of how to construct a visually-augmented HIN. There are three steps to construct a visually-augmented HIN given an original HIN containing only semantic factors:

Step 1: Image feature extraction: Given item images, an image feature extraction method is adopted to extract features in these images. These features are considered as visual information to be integrated into a HIN in the next step. Various feature extraction methods can be utilized depending on the specific types of image features preferred. In this work, five methods are considered.

Step 2: Visual factor nodes generation: After obtaining image features of item images, visual factor nodes are generated. These nodes are defined as representatives of significant image features extracted from item images and obtained by using clustering the extracted features.

Step 3: Visual relations integration: The last step is to integrate visual relations connecting between user/item and visual factor nodes in the original HIN. In this work, two types of visual relations are considered, i.e., user-visual factor relation type and item-visual factor relation type.

More details of these steps are discussed in the following section.

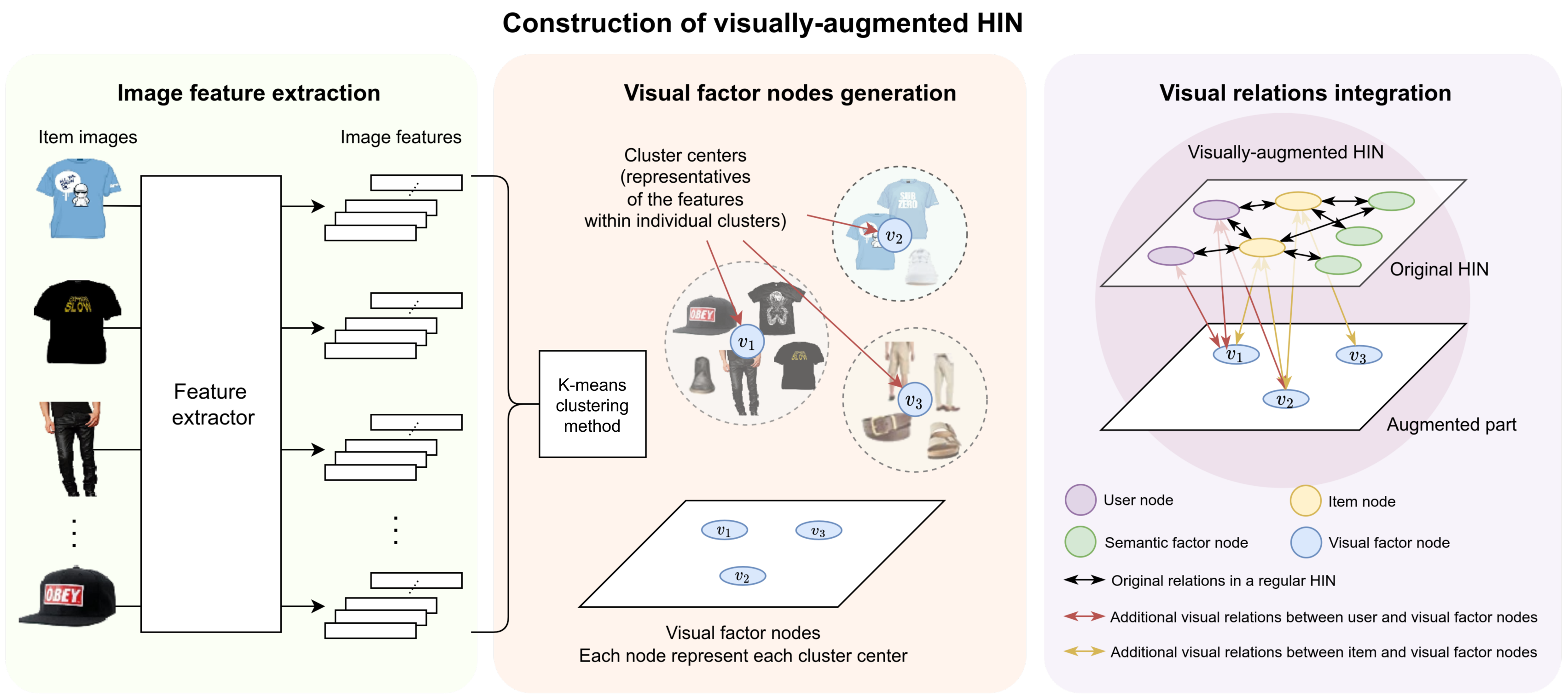

Construction of visually-augmented HIN
Image feature extraction
Item images
Image features
Feature extractor
Visual factor nodes generation
Cluster centers (representatives of the features within individual clusters)
K-means clustering method
v_1
v_2
v_3
Visual factor nodes
Each node represent each cluster center
Visual relations integration
Visually-augmented HIN
Original HIN
Augmented part
User node
Item node
Semantic factor node
Visual factor node
Original relations in a regular HIN
Additional visual relations between user and visual factor nodes
Additional visual relations between item and visual factor nodes

Figure 3.4: Overall framework of how to construct a visually-augmented HIN consisting of three steps: (1) image feature extraction, where visual features are extracted from item images using a selected method, (2) visual factor node generation, where features from the previous step are clustered using K-means clustering method to identify visual factor nodes, and (3) visual relations integration, where visual factor nodes previously generated are connected with user and item nodes in the original HIN via visual factor relations.

3.2.1 Image Feature Extraction

To construct a visually-augmented HIN, image features are first extracted from item images. Five image feature extraction methods are considered in this thesis, i.e., SIFT, SURF, ORB, Color Histogram, and a pre-trained CaffeNet model. Overall, SIFT, SURF, and ORB primarily focus on capturing information related to the texture, shape, and structure of visual elements within an image. They excel at detecting and describing distinctive features that exhibit specific patterns. These patterns include textures, corners, edges, and blobs. In contrast, color histograms capture the color information present in an image. Indeed, textures, shapes, and colors collectively cover a majority of the factors associated with users' visual preferences. However, considering that users' visual preferences can be intricate and extend beyond the capabilities of these individual feature extraction techniques, latent features extracted from CaffeNet are adopted. The rest of this thesis will refer to the latent features extracted from CaffeNet as "CNN features". This differentiation is intended to clearly highlight the difference between the latent features of images and the latent features of users/items obtained from recommendation models. The incorporation of CNN features serves to bridge the gap between the complexities of users' visual preferences and the limitations of the aforementioned feature extraction methods. By using all of these feature types, a broad range of visual information that likely corresponds to users' preferences is covered. Note that there are other more advanced image feature extraction methods, such as ResNet [189], GradCAM [190], and EfficientNet [191], that can be used for constructing visually-augmented HINs. However, this thesis focuses on these five feature extraction methods due to their simplicity in implementation and their usage in previous work on visually-aware recommender systems [37, 81, 92, 192].

3.2.2 Visual Factor Nodes Generation

In order to construct a visually-augmented HIN, visual factor nodes must be first generated. Visual factor nodes are representatives of significant image features extracted from item images. These image features can be of any type such as local keypoint descriptors from SIFT [25], SURF [26] or ORB [27], color histograms or hidden layer outputs from pre-trained deep learning models previously described. In this work, each feature extraction method is considered individually to generate visual factor nodes, without fusion of different types of visual features.

Given a dataset containing item images, the image features of all images are extracted using a selected feature extraction method. For SIFT, SURF, and ORB, multiple image feature vectors can be obtained per image. In the case of the color histogram and CNN features, one image feature vector is generated per image. These extracted feature vectors are then clustered into k_V clusters by using the K-means clustering method. Note that other clustering methods can also be applied. The K-means clustering method is selected due to its popularity and simplicity. After clustering, visual factors are defined as cluster centers of the extracted image features. Therefore, k_V cluster centers namely $\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{k_V}$ are obtained. Each of them is a vector that is a representative of image features (which are also vectors) within its cluster. Based on these visual factors, a set of visual factor nodes

$$\mathcal{V} = \{v_1, v_2, v_3, ..., v_{k_V}\} \tag{3.10}$$

is formed and added to $\mathcal{N}'$ where v_i is a visual factor node of visual factor $\mathbf{v}_i$ for every $i = 1, 2, 3, ..., k_V$.

3.2.3 Visual Relations Integration

After visual factors are generated, each item node is then connected to visual factor nodes depending on its image features. Specifically, for each image feature vector $\mathbf{f}$ extracted from an image of item p, its cluster is identified. Suppose that $\mathbf{f}$ belongs to cluster v_i. Then, the item node of p is connected to the visual factor node v_i in the visually-augmented HIN. In other words, r_{p,v_i} where $\psi(r_{p,v_i}) = R_{PV}$ and $r_{v_i,p}$ where $\psi(r_{v_i,p}) = R_{PV}^{-1}$ are added to $\mathcal{R}_V$. Note that this process is repeated for every image feature vector extracted from the image. Therefore, in the case that there are multiple image feature vectors extracted from the image, the item node corresponding to this image can be connected to multiple visual factors. For example, by using SIFT as a feature extraction method, multiple feature vectors can be obtained from an image of a given item p. Each of them may belong to a different cluster, i.e., it corresponds to a different visual factor. In this case, the item p node is connected to multiple visual factor nodes in the visually-augmented HIN.

Visual factors nodes are also connected to user nodes in order to facilitate users' visual preferences modeling. Given a user u, to connect the node of u to visual factor nodes, the user visual preference profile $\mathbf{v}^u$ must be computed first. Let $\mathcal{P}_u$ be the set of all u's items. $\mathbf{v}^u$ is computed by applying mean pooling [193] on all visual factors of his/her items in $\mathcal{P}_u$. Specifically, $\mathbf{v}^u$ is defined as:

$$\mathbf{v}^u = \sum_{p \in \mathcal{P}_u} \sum_{v_i \in \mathcal{V}_p} \mathbf{v}_i \tag{3.11}$$

where $\mathcal{V}_p$ is the set of visual factor nodes connected to item node p and $\mathbf{v}_i$ is the visual factor (which is a vector) corresponding to visual factor node v_i in $\mathcal{V}_p$.

Then, $\mathbf{v}^u$ is compared with every visual factor. The cosine similarity is used as a metric in this case. The set of top-k_s visual factors that are similar to user u is constructed. The relations with the types R_{UV} and R_{UV}^{-1} between u and each of the selected visual factors are added to $\mathcal{R}'$ where $\mathcal{R}'$ denotes the set of relations in a visually-augmented HIN. In other words, r_{u,v_i} where $\psi(r_{u,v_i}) = R_{UV}$ denoting a relation type connecting a user node to a visual factor node and $r_{v_i,u}$ where $\psi(r_{v_i,u}) = R_{UV}^{-1}$ denoting an inverse of R_{UV} are added to $\mathcal{R}_V$ for each visual factor v_i in the set of top-k_s most similar visual factors of user u.

3.3 Probabilistic Meta-Path

Regular meta-paths can represent only multi-hop relations that are static. For example, let U, P, and C denote the user, item, and category node types. A meta-path $UPCP$ suggests that a user may like an item only because it is in the same category as a user's previously interacted item. In some cases, users' preferences may depend on a mixture of factors. For instance, a user may prefer an item in the same category or an item that has a similar appearance as one of his/her items. Such combinations of multiple factors are called *hybrid factors*. To capture such preferences based on hybrid factors, this work uses a meta-path called *probabilistic meta-path* in which hybrid factors are considered based on pre-defined probabilities. It is defined as follows:

Definition 3.6. (Probabilistic Meta-Path) *Given a visually-augmented HIN $\mathcal{G}'$, a probabilistic meta-path*

$$m' = N_1 N_2 \cdots N_{i-1}\{\delta * N_i \oplus (1-\delta) * N_j\} N_{i+1} \cdots N_l \tag{3.12}$$

is defined as a sequence of node types, relation types, and their transition probability in schema $\mathbb{G}'$ of $\mathcal{G}'$ where $\oplus$ is a symbol that represents the "or" relation of the semantic node type N_i and the visual factor node type N_j and δ is a probability $0 \leq \delta \leq 1$. It contains at least one visual factor node type and one visual relation type. Starting from node type N_{i-1}, the next node type will go to semantic node type N_i with the probability δ and go to visual factor node type N_j with the probability $1-\delta$. For simplicity, the probability is ignored in the annotation. Therefore, m' can be denoted as $N_1 N_2 \cdots N_{i-1}\{N_i \oplus N_j\} N_{i+2} \cdots N_l$. The probability δ can be freely adjusted. When $\delta = 1$, visual factor node types will not be considered. They then *become regular meta-paths without hybrid factors involved. In other words, regular meta-paths can be considered as special cases of probabilistic meta-paths where the probability* $\delta = 1$.

Example 3.4. (Probabilistic Meta-Path) Figure 3.5 shows an example of using the regular meta-path $UPCP$ and the probabilistic meta-path $UP\{C \oplus V\}P$ with the probabilities of going to category node type (C) δ and visual factor node type (V) $1 - \delta$. The numbers and symbols on the edges indicate the probabilities assigned on those edges. By following the regular meta-path $UPCP$, a path instance ("User 1","T-shirt A","Category: T-shirt","T-shirt B") can be found. This suggests that "User 1" may like "T-shirt B" because it is in the same category as "T-shirt A". On the other hand, by following $UP\{C \oplus V\}P$, another path instance, ("User 1","T-shirt A","v_1","T-shirt B") can be found. It shows that "User 1" may also like "T-shirt B" because it has the same visual factor (v_1) as "T-shirt A". It can be seen that $UP\{C \oplus V\}P$ can reveal "User 1"'s preference in a more complex way compared to the regular meta-path $UPCP$.

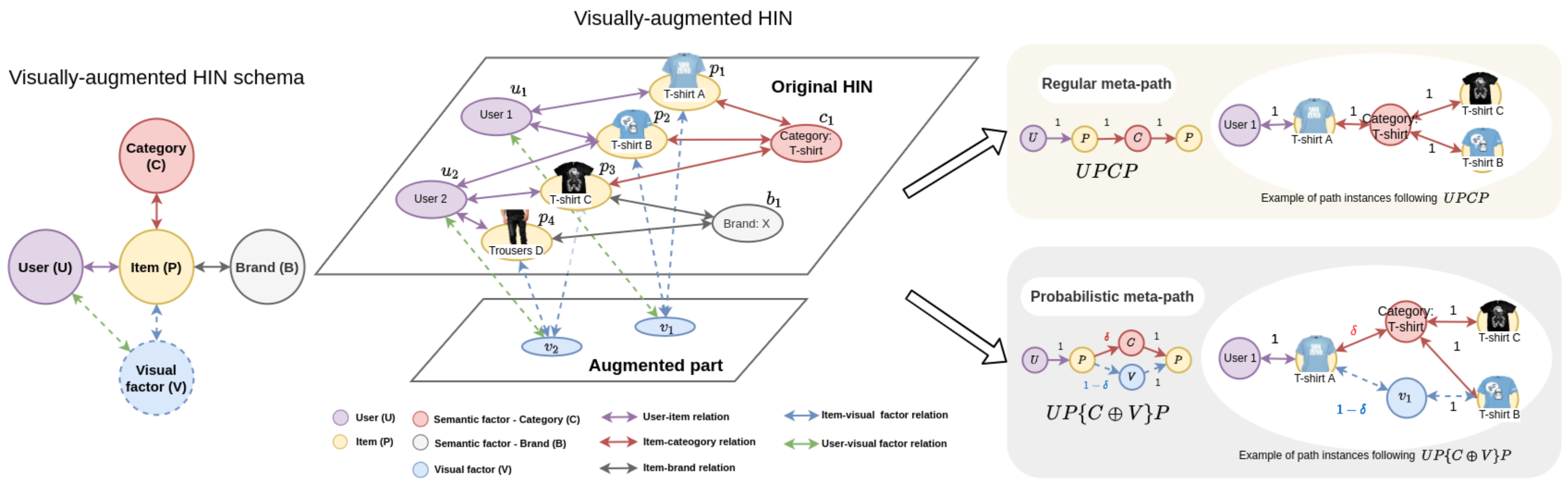


Figure 3.5: Example of visually-augmented HIN schema, visually-augmented HIN and the comparison of using the regular meta-path $UPCP$ and the probabilistic meta-path $UP\{C \oplus V\}P$ where U, P, C, and V denote the user, item, category, and visual factor node types, respectively. The probabilistic meta-path enables the discovery of more diverse multi-hop relations and reveals "User 1"'s preference in a more complex way compared to the regular meta-path.

3.4 The Proposed Recommendation Method Based on a Visually-Augmented HIN

Based on visually-augmented HINs, any graph-based recommendation models such as KGAT [5] can be applied to learn recommendations directly. However, learning latent representations or embedding of nodes has recently been proven to be effective for recommender systems and other applications [148, 174]. Therefore, this work proposes a visually-aware recommendation approach that uses embeddings of nodes in a visually-augmented HIN. This approach is based on a user-based CF-KNN model which is a simple yet effective recommendation model. Unlike the traditional CF-KNN model, the proposed approach uses node embeddings generated from a visually-augmented HIN to profile users instead of a user-item interaction matrix. To obtain node embeddings, the metapath2vec node embedding method is adopted. This method is applied with a visually-augmented HIN and a probabilistic meta-path to generate node embeddings comprising of visual information. This section provides a detailed description of the proposed approach.

The proposed recommender system is based on a user-based CF-KNN model and the metapath2vec method. The idea is to use the metapath2vec method to generate user representations based on a given visually-augmented HIN. Then, these user representations are used to predict recommendations using the CF-KNN method. For each user, to capture his/her personal preferences, his/her user representation is defined as a combination of the embeddings of his/her items in a visually-augmented HIN. Therefore, the first step is to generate item node embeddings in a given visually-augmented HIN. Node embedding methods using meta-paths including metapath2vec have been popularly used to generate node embeddings [133]. However, these approaches only consider semantic factors to form a neighborhood context. To consider both semantic and visual factors, a hybrid neighborhood context based on a probabilistic meta-path (defined in Section 3.6) is used instead. Given a probabilistic meta-path $m' = UN_1N_2 \cdots N_{i-1}\{\delta * N_i \oplus (1-\delta) * N_j\}N_{i+1} \cdots N_lP$ where U denotes a user node type, P denotes an item node type, N_i denotes a semantic node type, and N_j denotes a visual factor node type, a hybrid neighborhood context is defined as a set of item nodes that can be reached by meta-path based random walks based on m' starting from a given user node u. For example, Figure 3.6 illustrates an example of a visually-augmented HIN and a neighborhood context formed based on a regular meta-path $UPCP$ and a hybrid neighborhood context formed based on a probabilistic meta-path $UP\{C \oplus V\}P$. Suppose

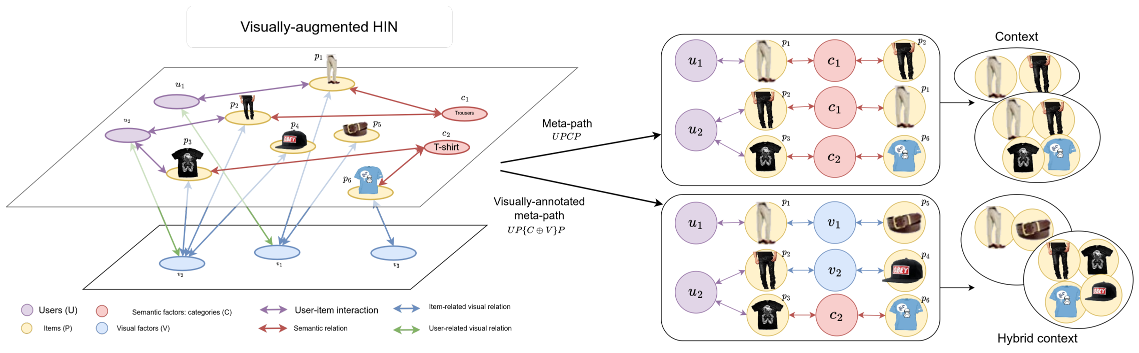


Figure 3.6: Example of a visually-augmented HIN and a neighborhood context formed based on a regular meta-path $UPCP$ and a hybrid neighborhood context formed based on a probabilistic meta-path $UP\{C \oplus V\}P$ where U, P, C, and V denote the user, item, category, and visual factor node types, respectively. This example illustrates how the hybrid context enables the inclusion of a broader range of items within the same context.

that each relation between any two nodes with the same relation type is equally important. A probabilistic meta-path $UP\{C \oplus V\}P$ forms a hybrid neighborhood context where those items that either have the same visual factor value or have the same category are considered similar. For item p_1, following this meta-path, item p_5 that has the same visual factor value as p_1 has the probability $1 - \delta$ of being put in the same context with p_1, while item p_2 that has the same category as p_1 has the probability δ of being put in the same context with p_1. Different from other previous work that only considers symmetric meta-paths, meta-paths used in the proposed framework are not restricted to only symmetric meta-paths, as the connectivity of different types of nodes is important in recommender systems [175].

After obtaining a hybrid neighborhood context based on a probabilistic meta-path, metapath2vec is applied to this context to generate item node embeddings of all items that appear in this context. Item node embeddings generated from metapath2vec are treated as item representations. These item representations capture similarities among items that belong in the same hybrid neighborhoods. They can be used as item features in content-based filtering recommender systems or used in an item-based CF-KNN model to predict recommendations. However, to leverage both information from the item side and user side, in this work, these item representations are used to compute user representations for a user-based CF-KNN model. Similarly to [193], each user representation is computed by using the mean of item node embeddings of this user's items. All user representations are then used to produce recommendations. The recommendation process is the same as the traditional user-based

CF-KNN model except the input is the generated user representations rather than a user-item interaction matrix.

After obtaining all user representations, the next step is to calculate similarity scores between users based on their representations. Several similarity metrics can be used, such as cosine similarity or Pearson correlation. In a CF-KNN model, the K-Nearest Neighbors algorithm is used to identify the K most similar users to the target user.

After that, the K-nearest neighbors are selected based on their similarity scores and consider only neighbors that have interacted with the same items the target user has not yet interacted with. Then, recommendations are generated for the target user based on the preferences of the selected neighbors. The system can either recommend items that the neighbors have rated highly or predict the recommendation scores of items towards the user. Specifically, given a user u and unobserved candidate item p, the recommendation score $\hat{x}_{up}$ can be determined by

$$\hat{x}_{up} = \sum_{u_k \in \mathcal{N}_u^p} sim\left(u, u_k\right) \tag{3.13}$$

where $\mathcal{N}_u^p$ is the set of u's K-nearest neighbors who have interacted with item p and $sim(u, u_k)$ denotes the similarity between user u and user u_k. Finally, the candidate items are ranked based on the computed scores and recommended to the target user.

3.5 Experimental Setup

The main objective of the experiments is to show the effectiveness of the proposed approach and how it performs when different visual features and different probabilistic meta-paths are applied in a top-N recommendation task. Also, since visually-augmented HINs are also applicable to many HIN-based recommender systems, the experiments were conducted to show the possibility and effectiveness of using visually-augmented HINs in a state-of-the-art HIN-based recommender system.

3.5.1 Datasets

The experiments were conducted on two datasets:

- **MovieLens dataset** [194, 195, 196, 197], an extension of the MovieLens dataset called HetRec2011-MovieLens-2K. It contains user tagging data, movie genres, actors, directors, and tags. As for visual information, movie posters were used as image data in this work. These movie posters were scraped from the OMDB [198] website and matched with the movie titles in the dataset. In these experiments, 5 node types and 10 relation types (inverse relation types included) were considered. These selected node and relation types are listed in Table 3.1.

- **Amazon dataset** [158, 199], consisting of users' reviews and item metadata in the "Clothing, Shoes and Jewelry" category. Only 5-rated reviews were retained in the dataset to ensure the users' satisfaction when learning their preferences. User ratings were converted to implicit feedback, i.e., 1 indicating that the user has rated the item and 0 indicating otherwise. For each item, its image link is provided in the dataset. These item images were downloaded via these links and used in the proposed approach. In these experiments, 4 node types and 8 relation types (inverse relation types included) were considered. The list of node and relation types used in the experiments is provided in Table 3.1.

Table 3.1: The statistics of MovieLens and Amazon datasets

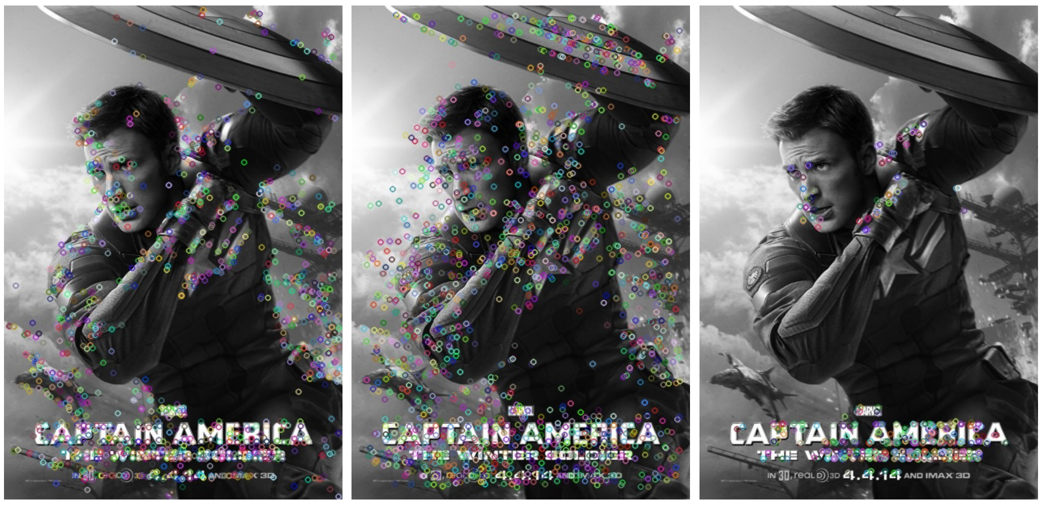
Dataset	Node type	#nodes	Relation type	#relations
MovieLens	user (U)	152	R_{UP}	3,870
	item (P)	301	R_{UV}	152
	genre (G)	18	R_{PG}	871
	tag (T)	3,031	R_{PT}	11,289
	visual factor (V)	100	R_{PV}(SIFT)	3,009
			R_{PV}(SURF)	3,009
			R_{PV}(ORB)	3,007
			R_{PV}(CH)	301
			R_{PV}(CNN)	301
Amazon	user (U)	12,491	R_{UP}	207,281
	item (P)	1,019	R_{UV}	12,491
	category (C)	604	R_{PC}	5,775
	visual factor (V)	100	R_{PV}(SIFT)	11,116
			R_{PV}(SURF)	1,979
			R_{PV}(ORB)	45,536
			R_{PV}(CH)	964
			R_{PV}(CNN)	964

For both datasets, to avoid the sparsity problem, 10-core data in which users and items have at least ten reviews or tagging records each were selected.

3.5.2 Parameterization and Experiment Environment

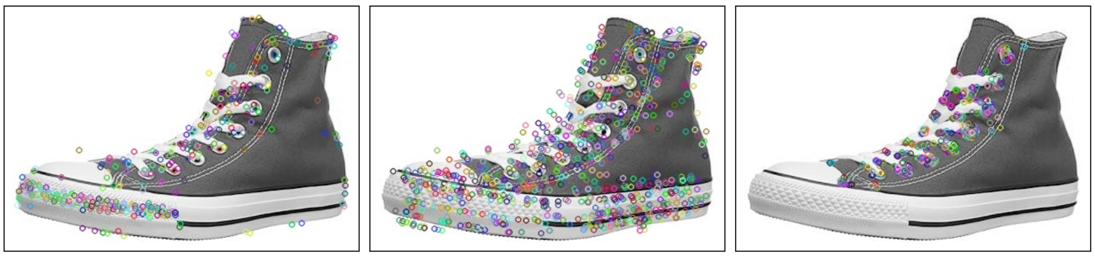
Visually-augmented HINs of both datasets were constructed based on five different visual feature types previously mentioned in Section 3.2. SIFT, SURF, and ORB features and Color Histograms were extracted by using *OpenCV* (Open Source Computer Vision Library) [200] which is an open-source computer vision and machine learning software library. It provides a comprehensive set of functions for image feature extraction written in C++ and supports multiple programming languages, including Python. For SIFT, there was no maximum number of keypoints set for each image. The initial value for the Gaussian blur that was applied to the image was set to 1.6. The number of layers per octave in the Gaussian pyramid was set to 3. The contrast threshold for keypoint detection was set to 0.04. The edge threshold for detecting poorly localized keypoints was set to 10. For SURF, the threshold value for the Hessian matrix response was set to 100. The number of octaves was set to 4. The number of layers per octave was set to 3. The size of descriptors was set to 64 dimensions. For ORB, the maximum number of keypoints was set to 500. The pyramid scale factor was set to 1.2. The number of pyramid levels in the image pyramid was set to 8. The threshold for the FAST corner detector was set to 20. The edge threshold was set to 31. The size of the patch used for computing the BRIEF descriptor was set to 31. CNN features were extracted from the second fully-connected layer, namely "FC7", of the pre-trained CaffeNet [201]. Each CNN feature is 4096-dimensional. It is worth noting that for SIFT, SURF, and ORB, there were multiple image features for each image depending on the number of its keypoints. Figures 3.7 and 3.8 display examples of keypoints detected by SIFT, SURF, and ORB on a movie poster image and a clothing item image, respectively. In these figures, each keypoint is marked by a circle. As shown in these examples, these feature extraction methods yield varying numbers of keypoints, consequently leading to different numbers of extracted image features. Meanwhile, there was only one Color Histogram and CNN feature extracted for each image.

For each feature type, the number of visual factors (k_V) was set to 100 (i.e., $K = 100$ in K-means clustering algorithm). The number of representative visual factors per user was set to 1 (i.e., $k^* = 1$). This parameter was chosen after comparing with $k^* = 3, 5$ and 10. They had similar accuracy but the selected setting required less computational time. The basic statistics of these HINs are shown in Table 3.1. To ensure that the hybrid contexts can



(a) SIFT (b) SURF (c) ORB

Figure 3.7: Examples of keypoints detected by (a) SIFT, (b) SURF, and (c) ORB on a movie poster image, with each keypoint indicated by a circle. These examples illustrate the varying numbers of keypoints detected by each method.



(a) SIFT (b) SURF (c) ORB

Figure 3.8: Examples of keypoints detected by (a) SIFT, (b) SURF, and (c) ORB on a clothing item image, with each keypoint indicated by a circle. These examples illustrate the varying numbers of keypoints detected by each method.

cover sufficient information, the number of generated paths for every starting node was set to 20. Some popularly used semantic meta-paths in literature [122] were selected to conduct experiments. The selected semantic meta-paths $m_{1,1}$, $m_{1,2}$, $m_{1,3}$ and $m_{1,4}$ are presented in Table 3.2. These meta-paths were used with regular HINs without visual factor nodes and visual relations. Meta-paths involving a visual factor node type V and probabilistic meta-paths were created based on these meta-paths. They are presented in Table 3.3. Note that meta-path $m_{2,1}$ is a regular meta-path that involves a visual factor node type. There is no probability involved in this meta-path. The purpose of using this meta-path is to compare with $m_{1,1}$ which is a similar semantic meta-path that does not consist of any visual factor

Table 3.2: The regular meta-paths selected for experiments

Meta-path	MovieLens dataset	Amazon dataset
$m_{1,1}$	UP	UP
$m_{1,2}$	$UPUPUP$	$UPUPUP$
$m_{1,3}$	$UPGPUP$	$UPCPUP$
$m_{1,4}$	$UPTPTP$	-

Table 3.3: The probabilistic meta-paths selected for experiments

Meta-path	MovieLens dataset	Amazon dataset
$m_{2,1}$	UVP	UVP
$m_{2,2}$	$U\{P \oplus V\}UP\{U \oplus V\}P$	$U\{P \oplus V\}UP\{U \oplus V\}P$
$m_{2,3}$	$UP\{G \oplus V\}P\{U \oplus V\}P$	$UP\{C \oplus V\}P\{U \oplus V\}P$
$m_{2,4}$	$UP\{T \oplus V\}P\{T \oplus V\}P$	-

node type. Meanwhile, $m_{2,2}$, $m_{2,3}$ and $m_{2,4}$ are probabilistic meta-paths. The first and the second probabilities in these probabilistic meta-paths were varied among $\{0, 0.25, 0.5, 0.75, 1\}$ to optimize the results. As for node embedding, the skip-gram method was applied with the size of embeddings set to 128 while the other settings were set as in [85]. For the user-based CF models, the number of user neighbors was set to 10 for all experiments. Regarding hardware settings and computing resources, a machine with a dual-core Intel(R) 1.80GHz CPU, an NVIDIA 16GB GPU, and 128GB RAM was employed.

3.5.3 Evaluation Metrics

The top-N recommendation performance was evaluated by three commonly used metrics, i.e., *Mean Precision@N* (Precision@N), *Mean Recall@N* (Recall@N), and *Mean F1 Score@N* (F1@N) with $N = 1, 5, 10, 50, 100$. These metrics are defined as follows:

$$Precision@N = \frac{1}{|\mathcal{U}|}\sum_{u \in \mathcal{U}} \frac{|\mathcal{P}_u \cap \mathcal{P}_u^N|}{N}, \tag{3.14}$$

$$Recall@N = \frac{1}{|\mathcal{U}|}\sum_{u \in \mathcal{U}} \frac{|\mathcal{P}_u \cap \mathcal{P}_u^N|}{|\mathcal{P}_u|}, \tag{3.15}$$

and

$$F1@N = 2 \cdot \frac{Precision@N \cdot Recall@N}{Precision@N + Recall@N} \tag{3.16}$$

where $\mathcal{U}$ is a set of users, $\mathcal{P}_u$ is the set of user u's items and $\mathcal{P}_u^N$ is the set of top-N recommended items of user u.

3.6 Results and Discussions

3.6.1 Comparison of the proposed approach and the baseline

In this section, the goal is to validate whether the proposed approach using visually-augmented HINs and probabilistic meta-paths is more effective than the approach using regular HINs and meta-paths in a top-N recommendation task. Therefore, the baseline model selected for comparison is a similar recommendation model using regular HINs and meta-paths. The proposed approach and the baseline chosen for this experiment are as follows:

- **MR**: a user-based CF-KNN model using metapath2vec++ [85] with regular meta-paths. This approach is similar to the proposed approach but visual factors are ignored. It is used to evaluate how effectively the proposed approach leverages visual information in visually-augmented HINs and produces recommendations based on this information. Similar to the proposed approach, metapath2vec++ was adopted to generate node embeddings of HINs without considering visual factor nodes and visual relations. Based on these embeddings, user representations were generated as in the proposed approach. Then, a CF-KNN model was used to produce recommendations based on these representations. The meta-paths used in this approach are in Table 3.2. Each meta-path was used to build a recommendation model. Other parameters including those in the metapath2vec++ method and a CF-KNN model were set as same as in the proposed approach.

- **VR**: the proposed approach using visually-augmented HINs and probabilistic meta-paths. This approach utilizes visual information integrated into visually-augmented HINs via probabilistic meta-paths and metapath2vec++ node embedding method. The meta-paths used in this approach are presented in Table 3.3. Similar to **MR**, each meta-path was used to build an individual recommendation model. For each model based on each meta-path, the parameters were fine-tuned, and the best result was selected for comparison.

First, **MR** and **VR** were compared when each pair of semantic meta-path $(m_{1,i})$ and its corresponding meta-path/probabilistic meta-path with a visual factor node type $(m_{2,i})$ was

Table 3.4: The selected parameters of **VR**-$m_{2,1}$, **VR**-$m_{2,2}$, **VR**-$m_{2,3}$, and **VR**-$m_{2,4}$ on **MovieLens** and **Amazon** datasets for comparisons. All probabilistic meta-paths ($m_{2,1}$, $m_{2,2}$, $m_{2,3}$, and $m_{2,4}$) are presented in Tables 3.3 (second column for **MovieLens** dataset and third column for **Amazon** dataset).

Dataset	Model	Meta-path probabilities	Visual feature type
MovieLens	**VR**-$m_{2,1}$	-	SURF
	VR-$m_{2,2}$	(0.75, 0.75)	ORB
	VR-$m_{2,3}$	(0.75, 0.25)	ORB
	VR-$m_{2,4}$	(0.25, 0.75)	CNN
Amazon	**VR**-$m_{2,1}$	-	SIFT
	VR-$m_{2,2}$	(0.0, 0.5)	ORB
	VR-$m_{2,3}$	(1.0, 1.0)	SURF

applied (for each $i = 1, 2, 3$ and 4). The parameters of **VR**-$m_{2,1}$, **VR**-$m_{2,2}$, **VR**-$m_{2,3}$, and **VR**-$m_{2,4}$ selected for comparison on this dataset are shown in Table 3.4. These parameters include probabilities in probabilistic meta-paths (denoted as a tuple where the first and the second elements are the first and the second probabilities where they are applicable respectively) and the visual feature type used for constructing the visually-augmented HIN. The parameters that gave the best result were selected for comparison.

Table 3.5 presents the F1@N results for both the baseline **MR** and the proposed approach **VR** on **MovieLens** and **Amazon** datasets, with the highest F1@N values in each column highlighted in bold. The results from this table suggest that the proposed approach **VR** outperformed the baseline approach **MR**, with **VR** models consistently achieving the highest F1@N values across various probabilistic meta-paths. Specifically, on **MovieLens** dataset, **VR**-$m_{2,2}$ performed best when $N = 1$ and 5, **VR**-$m_{2,4}$ performed best when $N = 10$, and **VR**-$m_{2,3}$ performed best when $N = 50$ and 100. Meanwhile, on **Amazon** dataset, **VR**-$m_{2,1}$ consistently demonstrated superior performance across all values of N.

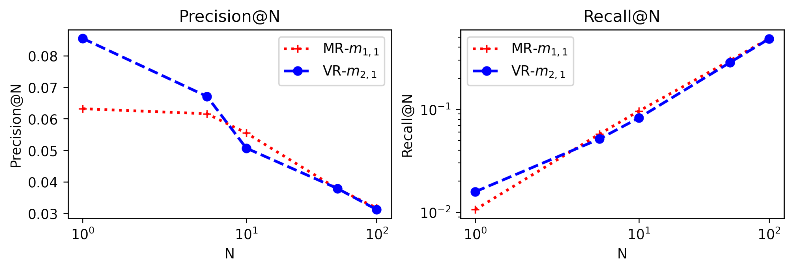
For a more in-depth analysis of the proposed approach's performance and comparisons with the baseline approach, the results of precision and recall are discussed next. Figure 3.9a, Figure 3.9b, Figure 3.9c, and Figure 3.9d show comparisons of **MR**-$m_{1,1}$ and **VR**-$m_{2,1}$, **MR**-$m_{1,2}$ and **VR**-$m_{2,2}$, **MR**-$m_{1,3}$ and **VR**-$m_{2,3}$, and **MR**-$m_{1,4}$ and **VR**-$m_{2,4}$ on **MovieLens** dataset respectively. From Figure 3.9a, **VR**-$m_{2,1}$ outperformed **MR**-$m_{1,1}$ in terms of both Precision@N

Table 3.5: F1@N results of the baseline **MR** and the proposed approach **VR** on **MovieLens** and **Amazon** datasets

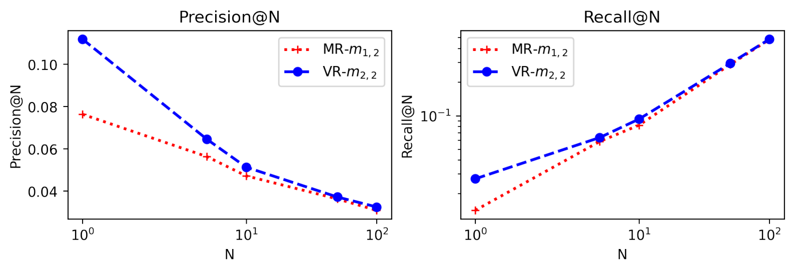
Dataset	Model	F1@1	F1@5	F1@10	F1@50	F1@100
MovieLens	**MR**-$m_{1,1}$	0.0182	0.0595	0.0703	0.0671	0.0595
	MR-$m_{1,2}$	0.0239	0.0575	0.0600	0.0645	0.0581
	MR-$m_{1,3}$	0.0249	0.0516	0.0662	0.0657	0.0585
	MR-$m_{1,4}$	0.0228	0.0524	0.0656	0.0656	0.0592
	VR-$m_{2,1}$	0.0267	0.0583	0.0627	0.0669	0.0586
	VR-$m_{2,2}$	**0.0438**	**0.0641**	0.0662	0.0660	0.0609
	VR-$m_{2,3}$	0.0378	0.0640	0.0650	**0.0737**	**0.0615**
	VR-$m_{2,4}$	0.0345	0.0608	**0.0710**	0.0714	0.0601
Amazon	**MR**-$m_{1,1}$	0.0383	0.0388	0.0283	0.0089	0.0072
	MR-$m_{1,2}$	0.0369	0.0383	0.0291	0.0138	0.0077
	MR-$m_{1,3}$	0.0364	0.0372	0.0259	0.0074	0.0053
	VR-$m_{2,1}$	**0.1024**	**0.0989**	**0.0670**	**0.0194**	**0.0126**
	VR-$m_{2,2}$	0.0618	0.0548	0.0389	0.0160	0.0089
	VR-$m_{2,3}$	0.0596	0.0556	0.0368	0.0104	0.0058

and Recall@N when $N = 1$. This indicates that, top-1 recommendations obtained from **VR**-$m_{2,1}$ are more accurate than those obtained from **MR**-$m_{1,1}$. **VR**-$m_{2,1}$ also outperformed **MR**-$m_{1,1}$ in terms of Precision@N when $N = 2$. For $N = 10, 50$ and 100, the proposed approach performed either slightly worse than **MR**-$m_{1,1}$ or equally as well as this baseline. One reason could be that, for this movie domain, relying on only visual factors by using meta-path UVP may not be as effective as using meta-path UP. However, the results of top-1 recommendations depict the potential of leveraging visual factors in the proposed approach by using meta-path UVP. Figure 3.9b shows that **VR**-$m_{2,2}$ performed better than **MR**-$M_{1,2}$ model in terms of Precision@N for every N. In terms of Recall@N, **VR**-$m_{2,2}$ performed better than **MR**-$M_{1,2}$ when $N = 1$. When N increases, their performance becomes similar. This result is similar to the result of **MR**-$m_{1,3}$ and **VR**-$m_{2,3}$ and **MR**-$m_{1,4}$ and **VR**-$m_{2,4}$. They both outperformed their corresponding baseline models in terms of Precision@N for every N and Recall@1 as can be seen in Figure 3.9c and Figure 3.9d. Considering all the results of the proposed models, it can be seen that **VR**-$m_{2,2}$, **VR**-$m_{2,3}$, and **VR**-$m_{2,4}$ performed similarly well in terms of both Precision@N and Recall@N. However, for further analysis, **VR**-$m_{2,2}$ was selected as the best model for this dataset as it achieved the highest Precision@N among the others. These results suggest that using probabilistic meta-paths to consider both semantic and visual factors is more effective than using regular meta-paths without considering visual factors.

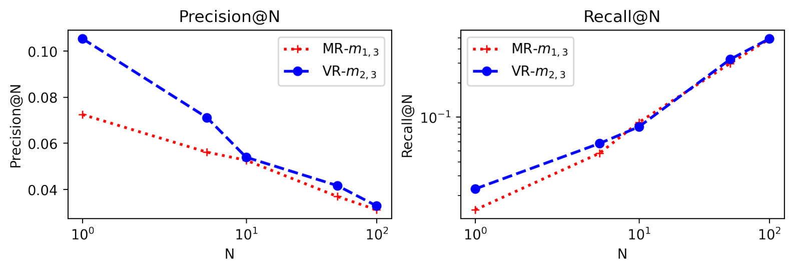
On **Amazon** dataset, comparisons of **MR**-$m_{1,1}$ and **VR**-$m_{2,1}$, **MR**-$m_{1,2}$ and **VR**-$m_{2,2}$, and **MR**-$m_{1,3}$ and **VR**-$m_{2,3}$ are shown in Figure 3.9a, Figure 3.9b, and Figure 3.9c respectively. Similar to **MovieLens** dataset, the parameters of **VR**-$m_{2,1}$, **VR**-$m_{2,2}$, and **VR**-$m_{2,3}$ selected for comparison on **Amazon** dataset are shown in Table 3.4. On **Amazon** dataset, **VR**-$m_{2,1}$ outperformed the baseline model **MR**-$m_{1,1}$ in terms of both Precision@N and Recall@N for every N as shown in Figure 3.9a. Compared to **MovieLens** dataset, **VR**-$m_{2,1}$ performed better on **Amazon** dataset. This means that relying on only visual factors by using meta-path $m_{2,1} = UVP$ is more effective on **Amazon** dataset than **MovieLens** dataset. This emphasizes that visual factors are more influential in the clothing domain than the movie domain. Considering probabilistic meta-paths involving hybrid factors, in Figure 3.9b, **VR**-$m_{2,2}$ outperformed the baseline model **MR**-$m_{1,2}$ in terms of both Precision@N and Recall@N for every N as well. In Figure 3.9c, **VR**-$m_{2,3}$ performed better than **MR**-$m_{1,3}$ in terms of Precision@N for every N and outperformed **MR**-$m_{1,3}$ in terms of Recall@N for $N = 1, 5, 10$ and 50. Comparing three **VR** models, i.e., **VR**-$m_{2,1}$, **VR**-$m_{2,2}$, and **VR**-$m_{2,3}$, on **Amazon** dataset, **VR**-$m_{2,1}$ performed best among these models. This suggests that using meta-path UVP is more effective than using $U\{P \oplus V\}UP\{U \oplus V\}P$ and $UP\{C \oplus V\}P\{U \oplus V\}P$ on this dataset. However, compared to those semantic meta-paths, using meta-paths that involve a visual factor node type is more effective. This indicates the effectiveness of the proposed approach using the visually-augmented HIN and the probabilities meta-paths on **Amazon** dataset.



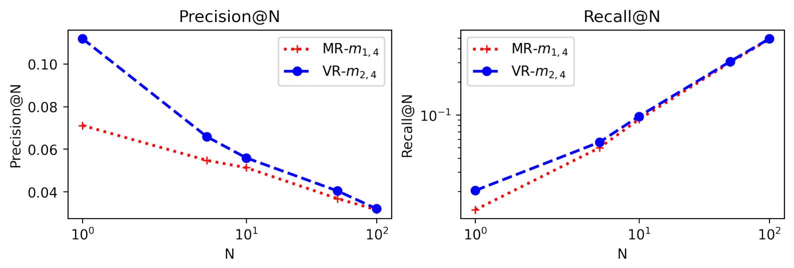
(a) **MR**-$m_{1,1}$ and **VR**-$m_{2,1}$



(b) **MR**-$m_{1,2}$ and **VR**-$m_{2,2}$

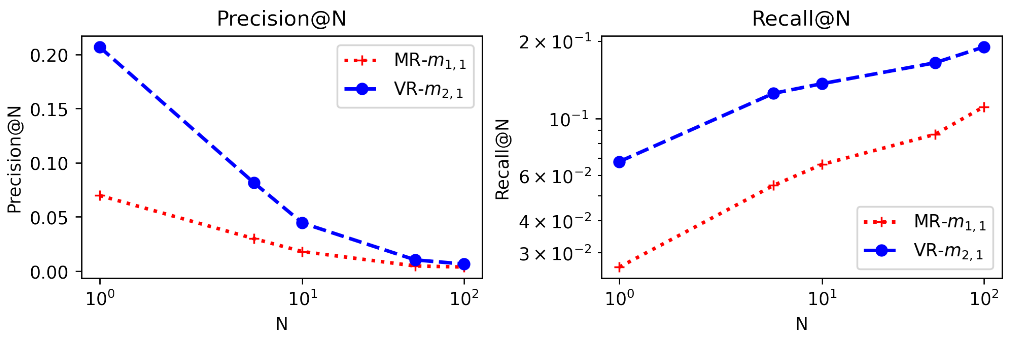


(c) **MR**-$m_{1,3}$ and **VR**-$m_{2,3}$

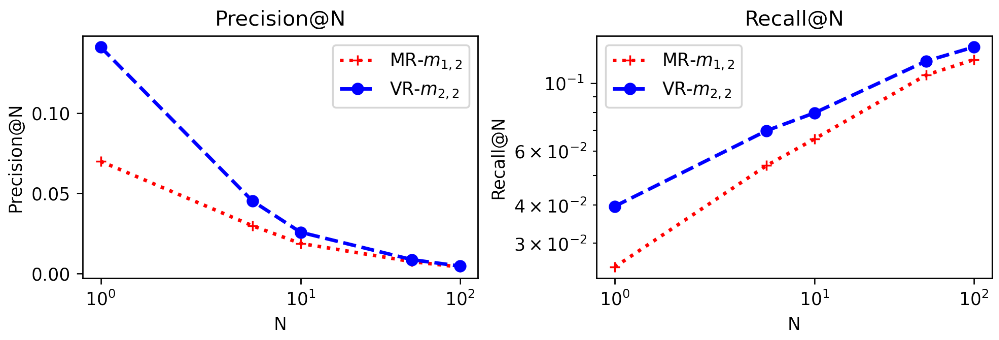


(d) **MR**-$m_{1,4}$ and **VR**-$m_{2,4}$

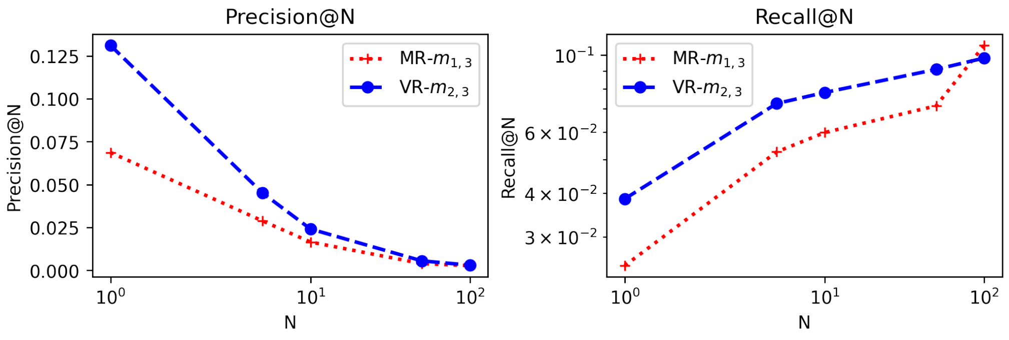
Figure 3.9: Comparisons of **MR** and **VR** when each pair of semantic meta-path ($m_{1,i}$) and its corresponding probabilistic meta-path ($m_{2,i}$) was applied (for $i = 1, 2, 3, 4$) on **MovieLens** dataset. All semantic meta-paths are presented in Table 3.2 (second column), and probabilistic meta-paths are presented in Table 3.3 (second column). These comparisons indicate that the proposed model **VR**, using a probabilistic meta-path, performed better than the baseline **MR**, using a similar meta-path without considering visual factors.



(a) **MR**-$m_{1,1}$ and **VR**-$m_{2,1}$



(b) **MR**-$m_{1,2}$ and **VR**-$m_{2,2}$



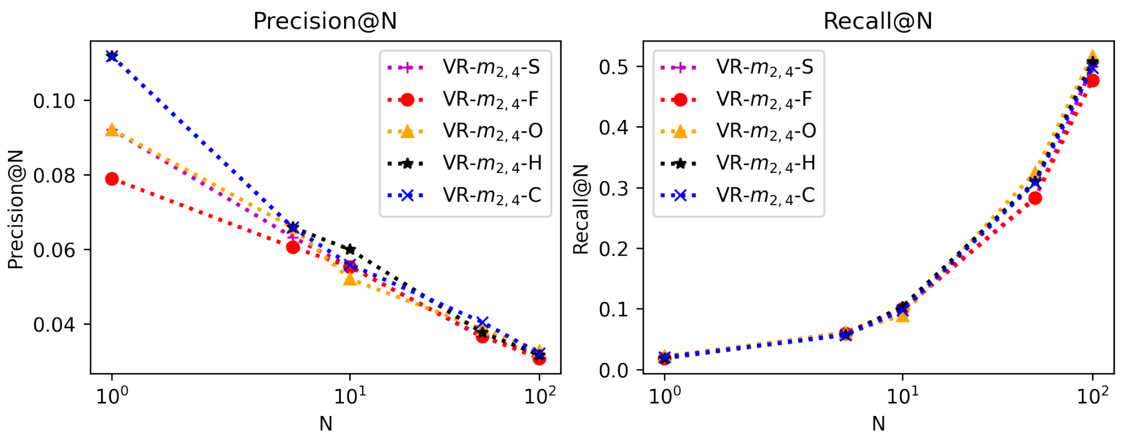
(c) **MR**-$m_{1,3}$ and **VR**-$m_{2,3}$

Figure 3.10: Comparisons of **MR** and **VR** when each pair of semantic meta-path ($m_{1,i}$) and its corresponding probabilistic meta-path ($m_{2,i}$) was applied (for $i = 1, 2, 3, 4$) on **Amazon** dataset. All semantic meta-paths are presented in Table 3.2 (third column), and probabilistic meta-paths are presented in Table 3.3 (third column). These comparisons indicate that the proposed model **VR**, using a probabilistic meta-path, performed better than the baseline **MR**, using a similar meta-path without considering visual factors.

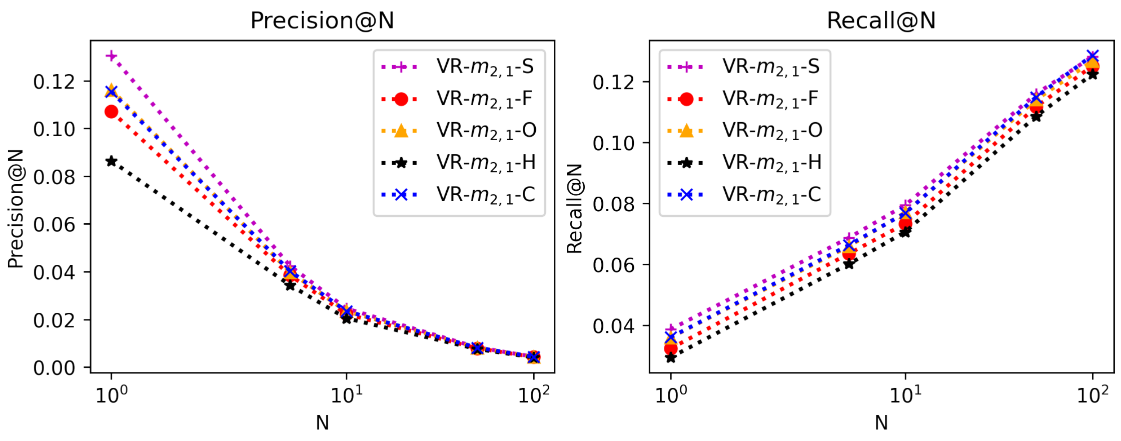
3.6.2 Comparison of different visual features

To discuss the effects of visual factor types, **VR** models using different types of visual factors, i.e., SIFT, SURF, ORB, Color Histogram, and CNN, were compared. For each dataset, the best meta-path was selected based on the results from the previous section. For **MovieLens**, the best meta-path is $m_{2,4}$. For **Amazon** dataset, the best meta-path is **VR**-$m_{2,1}$. Then, the **VR** approach with the selected meta-path was applied with each visual factor type for comparison. Figure 3.11 shows the results of **VR** models using different types of visual factors on **MovieLens** and **Amazon** datasets where **VR**-m_i-**S**, **VR**-m_i-**F**, **VR**-m_i-**O**, **VR**-m_i-**H** and **VR**-m_i-**C** denote the **VR** models using SIFT, SURF, ORB, Color Histogram and CNN visual factors with the meta-path m_i respectively.

On **MovieLens dataset**, the results are in Figure 3.11a. From this figure, **VR**-m_5-**H** and **VR**-m_5-**C** outperformed the other models in terms of Precision@N. Comparing these two models in terms of Precision@N, when $N = 1$ and 5, they both performed equally well. When $N = 10$, **VR**-m_5-**H** performed slightly better than **VR**-m_5-**C**. Meanwhile, when $N = 50$ and 100, **VR**-m_5-**C** performed better than **VR**-m_5-**H**. Despite the difference in Precision@N, in terms of Recall@N, all models performed similarly. Based on the results of Precision@N, it can be seen that Color Histogram and CNN features are more effective in capturing users' visual preferences on the movie posters compared to the other features. On **Amazon** dataset, **VR**-$m_{2,1}$-**S** performed better than the other models in terms of both Precision@N and Recall@N. The second-best model is **VR**-$m_{2,1}$-**C** using the CNN features extracted from item images. This model also performed similarly to **VR**-$m_{2,1}$-**O** using the ORB features. On the other hand, **VR**-$m_{2,1}$-**H** performed worse compared to the others. This suggests that texture and shape features are more effective in recognizing users' preferences compared to the color feature of items in this dataset.



(a) MovieLens



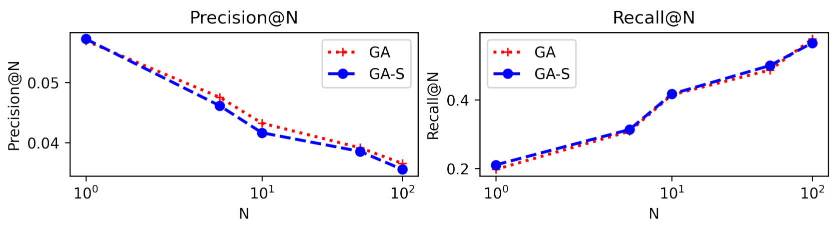
(b) Amazon

Figure 3.11: Accuracy results of the proposed approach **VR** with different visual factor types on (a) **MovieLens** dataset and (b) **Amazon** dataset where **VR**-m_i-**S**, **VR**-m_i-**F**, **VR**-m_i-**O**, **VR**-m_i-**H** and **VR**-m_i-**C** denote the **VR** models using SIFT, SURF, ORB, Color Histogram and CNN visual factors with the meta-path m_i respectively.

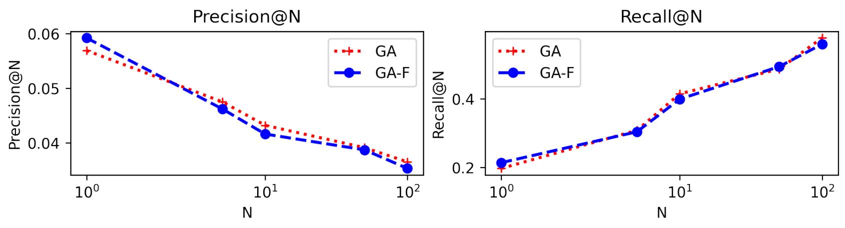
3.6.3 Effectiveness of visually-augmented HINs on a state-of-the-art HIN-based recommendation model

As previously mentioned, visually-augmented HINs can be applied with any HIN-based recommender system to leverage visual information within the networks for producing recommendations. To examine the potential of applying visually-augmented HINs with a HIN-based recommender system, a state-of-the-art HIN-based recommender system was applied with the constructed visually-augmented HINs of **MovieLens** and **Amazon** dataset. The state-of-the-art adopted in this experiment is the Knowledge Graph Attention Network (KGAT) [5] which combines a Graph Convolution Network (GCN) and a Graph Attention Network [128] to learn recommendations. For each dataset, KGAT was applied with the regular HIN without visual factor nodes and visual relations and six visually-augmented HINs based on different types of visual factors including SIFT, SURF, ORB, Color Histogram, and CNN visual factor types. The KGAT model applied with the regular HIN of each dataset is denoted as **GA**. Meanwhile, **GA-S**, **GA-F**, **GA-O**, **GA-H** and **GA-C** denote the KGAT models that were applied with the visually-augmented HINs containing SIFT, SURF, ORB, CH, and CNN visual factors respectively.

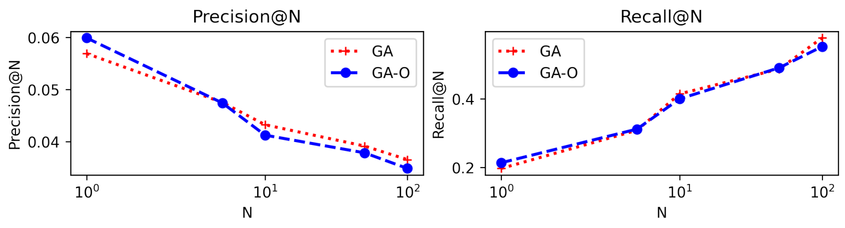
On **MovieLens** dataset, comparisons of **GA** and **GA-S**, **GA** and **GA-F**, **GA** and **GA-O**, **GA** and **GA-H**, and **GA** and **GA-C** are presented in Figure 3.12a, Figure 3.12b, Figure 3.12c, Figure 3.12d and Figure 3.12e respectively. From these figures, **GA-F**, **GA-O GA-H**, and **GA-C** performed better than **GA** in terms of both Precision@N and Recall@N when $N = 0$. When N increased, they all performed similarly to **GA** in terms of both metrics. This demonstrates that augmenting the original HIN with visual factor nodes and visual relations based on SURF, ORB, Color Histogram, and CNN features improved the accuracy of top-1 recommendations on this dataset. On **Amazon** dataset, comparisons of **GA** and **GA-S**, **GA** and **GA-F**, **GA** and **GA-O**, **GA** and **GA-H**, and **GA** and **GA-C** are shown in Figure 3.13a, Figure 3.13b, Figure 3.13c, Figure 3.13d, and Figure 3.13e respectively. It can be seen from these figures that most of the KGAT models applied with the visually-augmented HINs performed worse than the KGAT model applied with the original HIN. Especially for **GA-S**, **GA-F**, **GA-O**, and **GA-H**, these models underperformed **GA** in terms of both Precision@N and Recall@N. Only **GA-C** achieved a similar performance in terms of Precision@N with **GA**, but it still underperformed in terms of Recall@N. This suggests that, compared to **MovieLens** dataset, applying the KGAT model with the visually-augmented HINs of **Amazon** dataset is less effective. One reason is that, on **Amazon** dataset, there are already a large number of nodes and relations in the HIN. Including more nodes and relations may affect the effectiveness of the KGAT model. Further fine-tuning of parameters may be necessary to achieve higher accuracy. On the other hand, augmenting nodes and relations in the HINs of **MovieLens** dataset is more effective than **Amazon** dataset since it increases the chance to discover more relationships beyond those in the original HINs. It suggests that the augmentation may not work well in the KGAT model when the original HINs already contain a substantial number of nodes and relations.



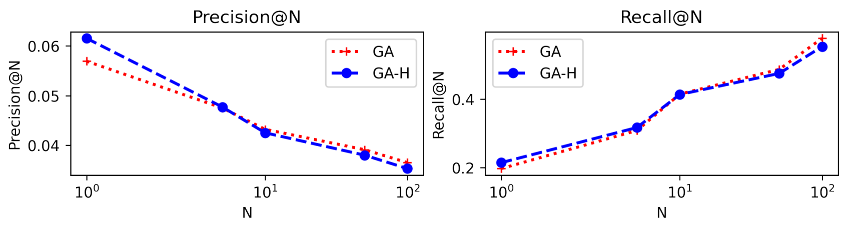
(a) **GA** and **GA-S**



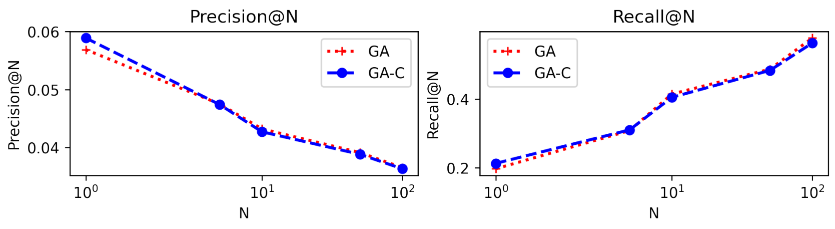
(b) **GA** and **GA-F**



(c) **GA** and **GA-O**

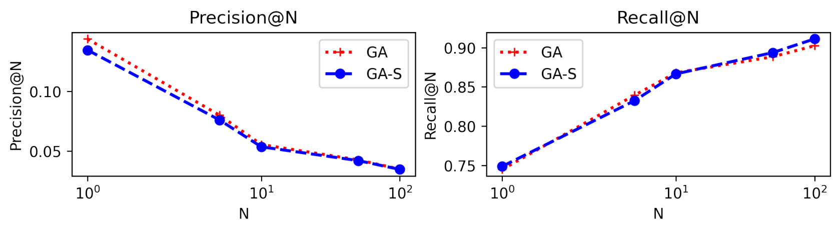


(d) **GA** and **GA-H**

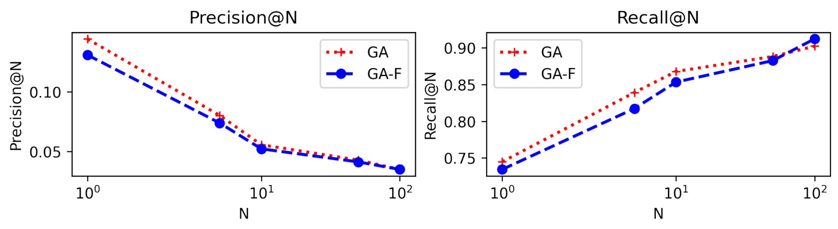


(e) **GA** and **GA-C**

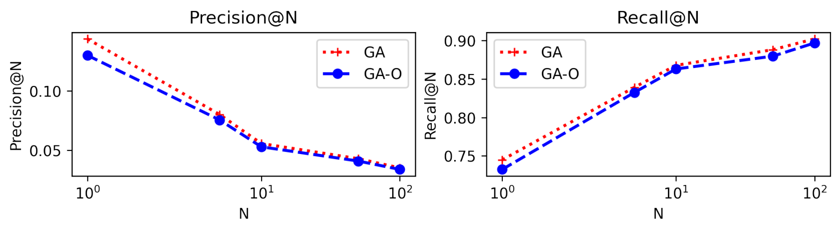
Figure 3.12: Comparisons of Knowledge Graph Attention Network (KGAT) [5], which is a state-of-the-art deep-learning based model, applied with the original HINs (**GA**) and KGAT applied with the visually-augmented HINs based on (a) SIFT (**GA-S**), (b) SURF (**GA-F**), (c) ORB (**GA-O**), (d) Color Histogram (**GA-H**), and (e) CNN features (**GA-C**) on **MovieLens** dataset



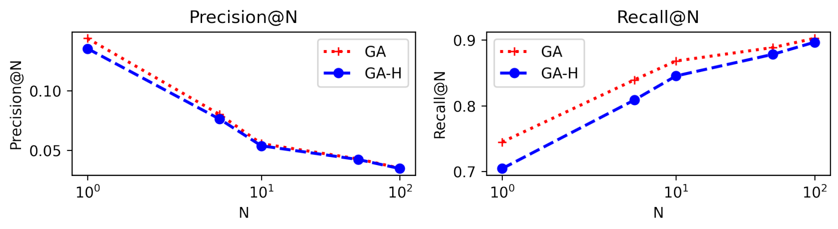
(a) **GA** and **GA-S**



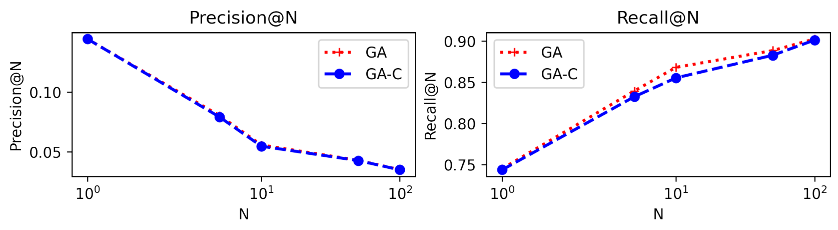
(b) **GA** and **GA-F**



(c) **GA** and **GA-O**



(d) **GA** and **GA-H**



(e) **GA** and **GA-C**

Figure 3.13: Comparisons of Knowledge Graph Attention Network (KGAT) [5], which is a state-of-the-art deep-learning based model, applied with the original HINs (**GA**) and KGAT applied with the visually-augmented HINs based on (a) SIFT (**GA-S**), (b) SURF (**GA-F**), (c) ORB (**GA-O**), (d) Color Histogram (**GA-H**), and (e) CNN features (**GA-C**) on **Amazon** dataset

3.7 Summary

In conclusion, this chapter addressed the first research question, which revolves around the simultaneous utilization of visual features and a HIN to develop visually-aware recommender systems. Specifically, it presented a method for integrating visual features into a HIN to enable visually-aware recommendations. At the beginning, this chapter introduced a visually-augmented HIN, which combines semantic information from user and item metadata with visual information from visual factor nodes and visual relations. This augmented HIN allowed for the learning of user latent representations that capture users' preferences from both semantic and visual perspectives. After that, a visually-aware recommendation approach based on visually-augmented HINs was proposed. The proposed approach addressed the gap in existing approaches by leveraging both semantic and visual information simultaneously in HIN-based recommendations. This was achieved by utilizing a hybrid context formed using probabilistic meta-paths, a novel type of meta-paths. These meta-paths facilitated the learning of user latent representations that captured both users' semantic and visual preferences. Finally, to generate visually-aware recommendations, the representations were applied with a user-based CF-KNN model, a simple and popularly used recommendation model.

Extensive experiments were conducted to evaluate the effectiveness of visually-augmented HINs and the recommendation approach based on them. The results demonstrated that, by leveraging visually-augmented HINs and the hybrid context, the proposed approach outperformed a similar approach that utilized only semantic information. This indicated the effectiveness of using visually-augmented HINs in accurately generating visually-aware recommendations.

Overall, this chapter contributes to the advancement of visually-aware recommendation systems by utilizing visually-augmented HINs and probabilistic meta-paths. The proposed approach offers the potential to leverage HINs in facilitating visually-aware recommendations, aligning with the first research question. However, the aspect of explainability has not been covered yet in this chapter. The following chapter will address the second research question regarding the explainability of recommender systems. It will provide a detailed description of how to build visually-aware recommender systems capable of generating accurate and explainable recommendations based on HINs.

Chapter 4

Explainable Visually-Aware Recommender Systems Using Meta-Paths

This chapter focuses on addressing the second research question (refer to Section 1.2), which pertains to the explainability of visually-aware recommendations using visually-augmented HINs. During the past decade, many studies have explored how to constrain recommender systems to produce explainable recommendations rather than non-explainable ones. Some approaches modified the traditional shallow recommendation models such as the MF model [203, 204] and the BPR-MF model [72]. In these approaches, the explainability scores of user-item pairs were considered as an additional soft constraint. These scores were often defined by using user/item neighborhoods [203] or association rules [72, 204]. Such definitions focus on only hop-1 relations (e.g., user-item interactions) and ignore rich information from multi-hop relations. Some attempts on using multi-hop relations to improve the explainability have been made [5, 61, 205, 206]. However, using multi-hop relations to improve the explainability may result in a scalability issue. These requirements in real-world situations have emphasized the importance of developing recommender systems capable of more than accurately predicting recommendations. Thus, how to design visually-aware recommender systems based on HINs with high accuracy, scalability, and explainability still needs to be explored.

Both scalability and explainability have been individually considered in developing visually-

This chapter has been published in [202] available at `https://doi.org/10.1016/j.knosys.2023.110258` aware recommender systems. Some attempts have been made to tackle a large-scale visually-aware recommendation problem [39, 207]. In terms of explainability, most existing work was proposed to enable image-based explainable recommendations [40, 45, 89]. However, visually-aware recommender systems that consider both scalability and explainability have still been rather overlooked. To bridge the gap of profiling users' individual visual preferences effectively and meet the performance requirements of Accuracy, Scalability, and Explainability, this chapter proposes a **S**calable and **E**xplainable **V**isually-aware **R**ecommender **S**ystem (SEV-RS) framework. Inspired by the omnipresence of heterogeneous information networks (HINs) [63] in explainable recommender systems, this work uses such networks to facilitate the task of visually-aware recommendation with explainability. However, many existing HIN-based recommendation frameworks typically ignored visual information and also suffered from scalability issues [122, 208]. Thus, unlike existing approaches, the proposed approach integrates visual elements into a HIN, extracts information from this HIN in a scalable way, and eventually uses this information to produce explainable visually-aware recommendations.

This proposed framework consists of three components. The first component is a **visually-augmented HIN**. This framework adopts a visually-augmented HIN and **probabilistic meta-paths**, previously introduced in the last chapter (see Chapter 3.2), as input for learning recommendations. It is worth noting that the first component does not constitute a unique contribution of this chapter. The major contribution of this chapter lies in the SEV-RS framework, particularly within its second and third components.

The second component is a **scalable meta-path feature extraction** method. Based on visually-augmented HIN and probabilistic meta-paths, meta-path features are extracted to profile users and items. Meta-path based approaches are popularly used to make HIN-based recommendations due to their capability of extracting semantically meaningful multi-hop relations [13]. However, it is challenging to develop effective and efficient methods for such approaches [5, 13, 85, 208]. For a length l meta-path, let n be the average number of adjacent nodes of every node in a HIN, the time cost for obtaining multi-hop relation is approximately n^l for each given starting node. Thus, leveraging such multi-hop relations of HINs may severely cause exponential time complexity and scalability issues [208, 209]. To alleviate such inefficiency, a scalable way to extract meta-path based features is proposed to profile each user and item. Depending on the meta-paths used in this method, different types of meta-path features can be extracted. By using meta-paths involving a visual factor node type, the extracted meta-path features can be used to facilitate visually-aware recommendations subsequently.

The third component is an **explainable recommendation generation** method. Meta-paths are capable of extracting meaningful multi-hop relations [13]. Due to this strength, they have been tremendously used to improve the explainability of recommendations [10]. This chapter introduces the concept of **meta-path based explainability** stemming from the proposed meta-path features. It allows us to quantify the "explainability" scores between user-item pairs based on a set of meta-paths. These scores can be leveraged in various recommender systems to provide explainability to these systems. However, since deep-learning based recommender systems usually suffer from scalability and also interpretability/explainability issues [6], this work proposes a shallow recommendation model that jointly considers the proposed meta-path features and the explainability factor to produce explainable visually-aware recommendations. Moreover, compared with deep-learning based recommender systems, the proposed model requires less computational time and is more scalable. In summary, this chapter provides the following contributions:

- A novel and scalable method for extracting meta-path features.
- A unique approach to quantifying explainability scores based on meta-paths between user-item pairs.
- A shallow recommendation model that combines the newly introduced meta-path features with the meta-path based explainability scores for learning explainable visually-aware recommendations.

The rest of this chapter is organized as follows: Section 4.1 explains how to efficiently generate user and item meta-path features based on visually-augmented HIN and probabilistic meta-paths. Section 4.2 revolves around the concept of meta-path based explainability and the computation of explainability scores for user-item pairs based on user and item meta-path features. Section 4.3 introduces the proposed approach for generating explainable visually-aware recommendations. Moving forward, Section 4.4 provides details of the experimental setup including datasets, parameterization, experiment environment, and evaluation metrics. Section 4.5 discusses the experimental results in the aspects of accuracy, explainability, and scalability. Finally, Section 4.6 concludes the chapter and summarizes the contributions.

4.1 Scalable Meta-Path Feature Extraction

Meta-paths have been used to determine the similarity (connectivity strength) between nodes in a HIN. Let u be a user, p be an item, m be a meta-path, and $\mathcal{Z}_m$ be a set of path instances of m connecting u and p. Let $s(u, p, m)$ denote the meta-path based connectivity strength of u and p following m. It can be calculated as the sum of the probabilities of path instances $z \in \mathcal{Z}_m$ [71]:

$$s(u, p, m) = \sum_{z \in \mathcal{Z}_m} Pr(z) \tag{4.1}$$

The higher the sum of the probabilities, the higher the connectivity strength. To achieve accurate recommendations, it is critical to find the most informative or predictive meta-paths. For user u, if the predictive meta-paths that lead to his/her observed items can be found, then it is more likely that these meta-paths will help find those unobserved items that he/she will be interested in. Intuitively, if the total connectivity strength between u and his/her observed items following m is high, then meta-path m is predictive/important for u. To measure the importance of a meta-path for a user, this work introduces the concept of *User-MetaPath association*.

Definition 4.1. (User-MetaPath association) *User-MetaPath association is the aggregated meta-path based connectivity strengths between u and his/her observed items following m. It is defined as $a_{u,m} = \sum_{p \in \mathcal{P}_u} s(u, p, m)$ where $\mathcal{P}_u$ is the set of observed items of u and $s(u, p, m)$ is the meta-path based connectivity strength between user u and item p following meta-path m.*

Similarly, the importance of a meta-path for an item can be measured. For an item p, if the total connectivity strength between p and its observed users denoted as $\mathcal{U}_p$ following meta-path m is high, then m is important to p. The concept of *Item-MetaPath association* is defined as follows.

Definition 4.2. (Item-MetaPath association) *Item-MetaPath association is the aggregated meta-path based connectivity strengths between p and its observed users following meta-path m. It is defined as $a_{p,m} = \sum_{u \in \mathcal{U}_p} s(u, p, m)$ where $\mathcal{U}_p$ is the set of users interacted with p and $s(u, p, m)$ is the meta-path based connectivity strength between user u and item p following meta-path m.*

Both User-MetaPath and Item-MetaPath associations can be computed from any meta-paths including probabilistic meta-paths. However, the connectivity strength is normally computed from a regular meta-path. Thus, this work proposes a novel method to compute the connectivity strength between user and item nodes based on a probabilistic meta-path. In this way, the most informative or predictive probabilistic meta-path for each user/item can be determined.

Given a probabilistic meta-path $m' = N_1N_2 \cdots N_{i-1}\{N_i \oplus N_j\}N_{i+2} \cdots N_l$, a path instance can follow either $m'_S = N_1N_2 \cdots N_{i-1}N_iN_{i+2} \cdots N_l$ with the probability δ, or $m'_V = N_1N_2 \cdots N_{i-1}N_jN_{i+2} \cdots N_l$ with the probability $1 - \delta$. Thus, the connectivity strength between user u and item p following m' is defined as the weighted sum of connectivity strength following m'_S and m'_V as follows:

$$s(u,p,m') = \delta \cdot \sum_{z \in \mathcal{Z}_{m'_S}} Pr(z) + (1-\delta) \cdot \sum_{z \in \mathcal{Z}_{m'_V}} Pr(z) \tag{4.2}$$

where $\mathcal{Z}_{m'_S}$ and $\mathcal{Z}_{m'_V}$ are sets of path instances of m'_S and m'_V respectively.

Computing Eq. (4.2) requires all path instances in $\mathcal{Z}_{m'_S}$ and $\mathcal{Z}_{m'_V}$ which is not scalable. To address this issue, this work proposes a novel scalable approach to compute $s(u,p,m')$. Inspired by the processing of a sentence (i.e., a sequence of words) in Natural Language Processing, the *first-order Markov assumption* is applied to calculate $Pr(z)$. For any z, it is assumed that the probability of each node in z depends only on its previous nodes. Thus, $Pr(z)$ can be computed by

$$Pr(z) = Pr(u, n_1, n_2, \cdots, n_l, p) = Pr(u)Pr(n_1|u)Pr(n_2|n_1) \cdots Pr(p|n_l) \tag{4.3}$$

where $Pr(u) = \frac{c_u}{|\mathcal{N}|}$ is the probability of node u in a HIN where c_u is the total number of user u nodes in a HIN and $|\mathcal{N}|$ is the total number of nodes in a HIN, and $Pr(y|x)$ is the probability of node y given x as a previous node in z for any $x, y \in \{u, n_1, n_2, ..., n_l, p\}$ computed by

$$Pr(y|x) = \frac{w(x,y)}{\sum_{n \in \mathcal{N}} w(x,n)} \tag{4.4}$$

Note that, for every user u, $Pr(u)$ is constant since every user has only one node in a HIN (i.e., $c_u = 1$ for every user u) and the total number of nodes in a HIN is constant. Thus, this term is ignored in the connectivity computation.

Considering $\sum_{z \in \mathcal{Z}_{m'_S}}$ and $\sum_{z \in \mathcal{Z}_{m'_V}}$ in Eq. (4.2), they are equivalent to the summations over all possible combinations of node sequences following m'_S and m'_V respectively. Thus, these two summations can be replaced by the series of summations that consider all combinations of node sequences following m'_S and m'_V instead as follows:

$$\sum_{z \in \mathcal{Z}_{m'_S}} Pr(z) = \sum_{n_1 \in \mathcal{N}_1} \sum_{n_2 \in \mathcal{N}_2} \cdots \sum_{n_i \in \mathcal{N}_i} \cdots \sum_{n_l \in \mathcal{N}_l} Pr(n_1|u)Pr(n_2|n_1) \cdots Pr(p|n_l)$$

$$= \sum_{n_1 \in \mathcal{N}_1} \sum_{n_l \in \mathcal{N}_l} Pr(n_1|u) \sum_{n_2 \in \mathcal{N}_2} \cdots \sum_{n_{l-1} \in \mathcal{N}_{l-1}} Pr(n_2|n_1) \cdots Pr(n_l|n_{l-1}) Pr(p|n_l) \tag{4.5}$$

and

$$\sum_{z \in \mathcal{Z}_{m'_V}} Pr(z) = \sum_{n_1 \in \mathcal{N}_1} \sum_{n_2 \in \mathcal{N}_2} \cdots \sum_{n_j \in \mathcal{N}_j} \cdots \sum_{n_l \in \mathcal{N}_l} Pr(n_1|u) Pr(n_2|n_1) \cdots Pr(p|n_l)$$

$$= \sum_{n_1 \in \mathcal{N}_1} \sum_{n_l \in \mathcal{N}_l} Pr(n_1|u) \sum_{n_2 \in \mathcal{N}_2} \cdots \sum_{n_{l-1} \in \mathcal{N}_{l-1}} Pr(n_2|n_1) \cdots Pr(n_l|n_{l-1}) Pr(p|n_l) \tag{4.6}$$

where $\mathcal{N}_k$ is the set of nodes of N_k type ($k = 1, 2, ..., l$). Both Eq. (4.5) and (4.6) can be computed similarly. Therefore, for simplicity, $\sum_{z \in \mathcal{Z}_{m'_S}} Pr(z)$ in Eq. (4.5) is first considered. Let n be the average number of adjacent nodes per node, computing Eq. (4.5) requires time complexity of $O(n^l)$, which is computationally expensive. Therefore, an alternative way is proposed to reduce the computational time by estimating the following term:

$$\sum_{n_2 \in \mathcal{N}_2} \cdots \sum_{n_{l-1} \in \mathcal{N}_{l-1}} Pr(n_2|n_1) \cdots Pr(n_l|n_{l-1}).$$

This term represents the connectivity from n_1 to n_l. This can be considered as *local connectivity* since it considers the relations between some particular nodes at the path-instance level. For example, given a visually-augmented HIN in Figure 4.1, the purple curved dashed box in Figure 4.2 represents the local connectivity between "T-shirt B" and "Category: T-shirt".

Computing the local connectivity from n_1 to n_l in each path instance is time-consuming. Instead of considering the connectivity between nodes at the path-instance level, the connectivity between node types at the meta-path level can bed used to measure the importance of a meta-path. This connectivity is called *global connectivity* of a meta-path m denoted as $g(m)$. It is computed by

$$g(m) = \prod_{k=1}^{l-1} C(k, k+1), \tag{4.7}$$

where $C(k, k+1)$ denotes the probability of N_{k+1} type nodes given N_k type nodes computed by

$$C(k, k+1) = \frac{\sum_{n' \in \mathcal{N}_k} \sum_{n'' \in \mathcal{N}_{k+1}} w(n', n'')}{\sum_{n' \in \mathcal{N}_k} \sum_{n \in \mathcal{N}} w(n', n)} \tag{4.8}$$

where $w(n', n'')$ and $w(n', n)$ denote the weights of the relations from n' to n'' and from n' to n respectively. Each $C(k, k+1)$ indicates the connectivity between one node type to another node type. Considering them all, $g(m)$ therefore indicates the connectivity between general N_1 type nodes to N_l type nodes through $N_2, ..., N_{l-1}$. Without actual path instances,

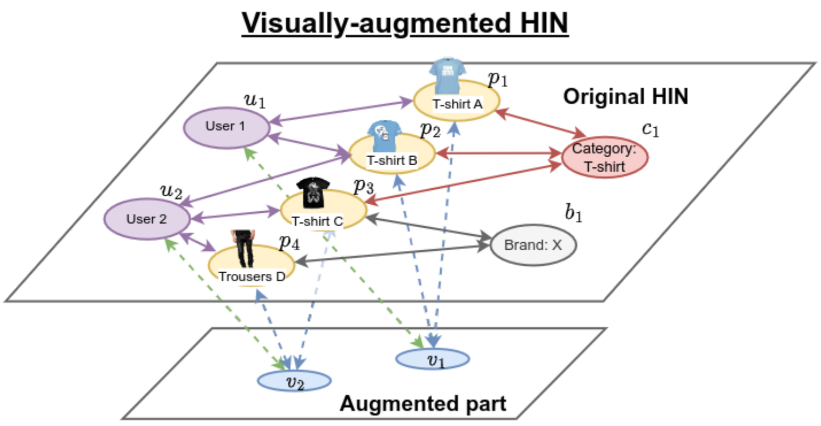


Figure 4.1: Example of a visually-augmented HIN. In this augmented HIN, there are five node types, i.e., user, item, category, brand, and visual factor node types.

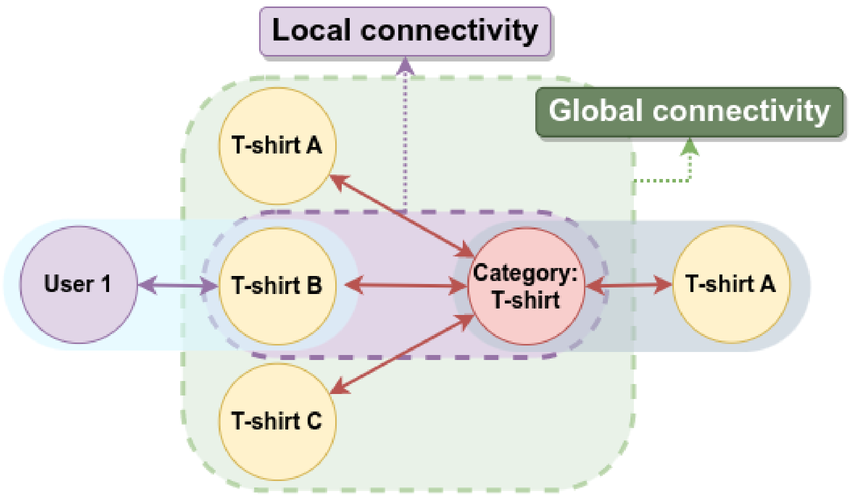


Figure 4.2: Example of local connectivity and global connectivity. The local connectivity, in a purple rounded rectangle, signifies the connectivity between "T-shirt B" and "Category: T-shirt" at the path-instance level. The global connectivity, in a green rounded rectangle, indicates the connectivity between item and category node types at the meta-path level.

$g(m)$ is used to find how likely user u links to item p following meta-path m. This global connectivity is shown as the green curved dashed box in Figure 4.2. From this figure, it can be seen that the global connectivity measures the general connectivity between overall item nodes and overall category nodes, rather than the specific connectivity between one/some item nodes and one/some category nodes. After substituting the local connectivity with the global connectivity in Eq. (4.5), Eq. (4.5) can be rewritten as:

$$\sum_{z \in \mathcal{Z}_{m'_S}} Pr(z) = g(u, m'_S) g(m'_S) g(m'_S, p) \tag{4.9}$$

where

$$g(u, m'_S) = \sum_{n_1 \in \mathcal{N}_1} Pr(n_1|u) \tag{4.10}$$

and

$$g(m'_S, p) = \sum_{n_l \in \mathcal{N}_l} Pr(p|n_l) \tag{4.11}$$

Similarly, $\sum_{z \in \mathcal{Z}_{m'_V}} Pr(z)$ in Eq. (4.6) is computed as follows:

$$\sum_{z \in \mathcal{Z}_{m'_V}} Pr(z) = g(u, m'_V) g(m'_V) g(m'_V, p) \tag{4.12}$$

Hence, $s(u, p, m')$ is computed as follows:

$$s(u, p, m') = \delta \cdot g(u, m'_S) g(m'_S) g(m'_S, p) + (1 - \delta) \cdot g(u, m'_V) g(m'_V) g(m'_V, p) \tag{4.13}$$

Figure 4.3 illustrates an example of how to compute the meta-path based connectivity strength of "User 1" and "T-shirt A" following a probabilistic meta-path $m' = UP\{C \oplus V\}P$ given a visually-augmented HIN shown in Figure 4.1. This connectivity strength is used to compute User-MetaPath association $a_{u,m'}$ and Item-MetaPath association $a_{p,m'}$.

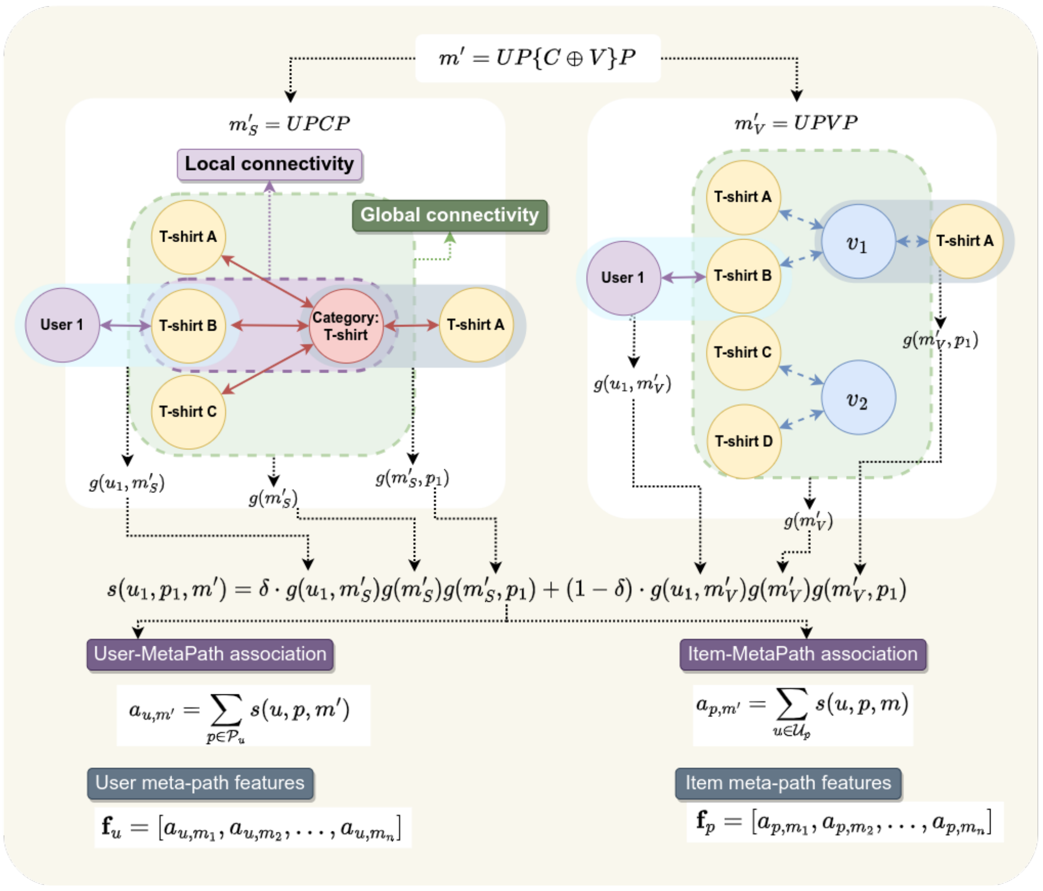

$m' = UP\{C \oplus V\}P$
$m'_S = UPCP$
Local connectivity
Global connectivity
T-shirt A
User 1
T-shirt B
Category: T-shirt
T-shirt A
T-shirt C
$g(u_1, m'_S)$
$g(m'_S)$
$g(m'_S, p_1)$
$m'_V = UPVP$
T-shirt A
v_1
T-shirt A
User 1
T-shirt B
T-shirt C
v_2
T-shirt D
$g(m'_V, p_1)$
$g(u_1, m'_V)$
$g(m'_V)$
$s(u_1, p_1, m') = \delta \cdot g(u_1, m'_S)g(m'_S)g(m'_S, p_1) + (1-\delta) \cdot g(u_1, m'_V)g(m'_V)g(m'_V, p_1)$
User-MetaPath association
$a_{u,m'} = \sum_{p \in \mathcal{P}_u} s(u, p, m')$
Item-MetaPath association
$a_{p,m'} = \sum_{u \in \mathcal{U}_p} s(u, p, m)$
User meta-path features
$\mathbf{f}_u = [a_{u,m_1}, a_{u,m_2}, \ldots, a_{u,m_n}]$
Item meta-path features
$\mathbf{f}_p = [a_{p,m_1}, a_{p,m_2}, \ldots, a_{p,m_n}]$

Figure 4.3: Example computation of the meta-path based connectivity strength of "User 1" and "T-shirt A" following a probabilistic meta-path $m' = UP\{C \oplus V\}P$, User-MetaPath association, and Item-MetaPath association. The probabilistic meta-path is split into two meta-paths. The global connectivity values based on each meta-path are calculated and combined to compute the User-MetaPath association and Item-MetaPath association. These associations, in turn, form user and item meta-path features.

Example 4.1. (User-MetaPath Association) Given the visually-augmented HIN in Figure 4.1, let all relations have the same weight $w(x, y) = w(y, x) = 1$. Let $\mathcal{P}$ denote the set of item nodes, $\mathcal{C}$ denote the set of category nodes and $\mathcal{B}$ denote the set of brand nodes. Given a probabilistic meta-path $m'_1 = UP\{C \oplus V\}P$ and $\delta = 0.4$, u_1's User-MetaPath association is computed by

$$a_{u_1,m'_1} = \sum_{p \in \mathcal{P}_{u_1}} s(u_1, p, m'_1) = s(u_1, p_1, m'_1) + s(u_1, p_2, m'_1) \tag{4.14}$$

where

$$s(u_1, p_1, m'_1) = \delta \cdot g(u_1, m'_{1S})g(m'_{1S})g(m'_{1S}, p_1) + (1 - \delta) \cdot g(u_1, m'_{1V})g(m'_{1V})g(m'_{1V}, p_1)$$

and

$$s(u_1, p_2, m'_1) = \delta \cdot g(u_1, m'_{1S})g(m'_{1S})g(m'_{1S}, p_2) + (1 - \delta) \cdot g(u_1, m'_{1V})g(m'_{1V})g(m'_{1V}, p_2)$$

where

$$\begin{aligned} g(u_1, m'_{1S}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_1) \\ &= Pr(p_1|u_1) + Pr(p_2|u_1) + Pr(p_3|u_1) + Pr(p_4|u_1) + Pr(p_5|u_1) \\ &= 1/3 + 1/3 + 0/3 + 0/3 + 0/3 = 2/3, \end{aligned}$$

$$\begin{aligned} g(u_1, m'_{1V}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_1) \\ &= Pr(p_1|u_1) + Pr(p_2|u_1) + Pr(p_3|u_1) + Pr(p_4|u_1) + Pr(p_5|u_1) \\ &= 1/3 + 1/3 + 0/3 + 0/3 + 0/3 = 2/3, \end{aligned}$$

$$g(m'_{1S}, p_1) = \sum_{n_2 \in \mathcal{C}} Pr(p_1|n_2) = Pr(p_1|c_1) = 1/3,$$

$$g(m'_{1V}, p_1) = \sum_{n_2 \in \mathcal{V}} Pr(p_1|n_2) = Pr(p_1|v_1) + Pr(p_1|v_2) = 1/3 + 0/3 = 1/3,$$

$$g(m'_{1S}, p_2) = \sum_{n_2 \in \mathcal{C}} Pr(p_2|n_2) = Pr(p_2|c_1) = 13,$$

$$g(m'_{1V}, p_2) = \sum_{n_2 \in \mathcal{V}} Pr(p_2|n_2) = Pr(p_2|v_1) + Pr(p_2|v_2) = 1/3 + 0/3 = 1/3,$$

$$g(m'_{1S}) = C(P, C) = 3/14$$

$$g(m'_{1V}) = C(P, V) = 4/14,$$

where 3 is the total number of relations from P to C, 4 is the total number of relation from P to V and 14 is the total number of relations from P to any type including the additional visual relation type. Thus

$$s(u_1, p_1, m'_1) = 0.4 \cdot (2/3)(3/14)(1/3) + (0.6) \cdot (2/3)(4/14)(1/3) \approx 0.06$$

and

$$s(u_1, p_2, m'_1) = 0.4 \cdot (2/3)(3/14)(1/3) + (0.6) \cdot (2/3)(4/14)(1/3) \approx 0.06$$

Hence,

$$a_{u_1,m'_1} = s(u_1, p_1, m'_1) + s(u_1, p_2, m'_1) \approx 0.12. \tag{4.15}$$

Similarly, for $m'_2 = UP\{B \oplus V\}P$, a_{u_1,m'_2} is calculated as follows:

$$a_{u_1,m'_2} = \sum_{p \in \mathcal{P}_{u_1}} s(u_1, p, m'_2) = s(u_1, p_1, m'_2) + s(u_1, p_2, m'_2)$$

where

$$s(u_1, p_1, m'_2) = \delta \cdot g(u_1, m'_{2S}) g(m'_{2S}) g(m'_{2S}, p_1) + (1 - \delta) \cdot g(u_1, m'_{2V}) g(m'_{2V}) g(m'_{2V}, p_1)$$

and

$$s(u_1, p_2, m'_2) = \delta \cdot g(u_1, m'_{2S}) g(m'_{2S}) g(m'_{2S}, p_2) + (1 - \delta) \cdot g(u_1, m'_{2V}) g(m'_{2V}) g(m'_{2V}, p_2)$$

where

$$\begin{aligned} g(u_1, m'_{2S}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_1) \\ &= Pr(p_1|u_1) + Pr(p_2|u_1) + Pr(p_3|u_1) + Pr(p_4|u_1) + Pr(p_5|u_1) \\ &= 1/3 + 1/3 + 0/3 + 0/3 + 0/3 = 2/3, \end{aligned}$$

$$\begin{aligned} g(u_1, m'_{2V}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_1) \\ &= Pr(p_1|u_1) + Pr(p_2|u_1) + Pr(p_3|u_1) + Pr(p_4|u_1) + Pr(p_5|u_1) \\ &= 1/3 + 1/3 + 0/3 + 0/3 + 0/3 = 2/3, \end{aligned}$$

$$g(m'_{2S}, p_1) = \sum_{n_2 \in \mathcal{B}} Pr(p_1|n_2) = Pr(p_1|b_1) = 0,$$

$$g(m'_{2V}, p_1) = \sum_{n_2 \in \mathcal{V}} Pr(p_1|n_2) = Pr(p_1|v_1) + Pr(p_1|v_2) = 1/3 + 0/3 = 1/3,$$

$$g(m'_{2S}, p_2) = \sum_{n_2 \in \mathcal{B}} Pr(p_2|n_2) = Pr(p_2|b_1) = 0,$$

$$g(m'_{2V}, p_2) = \sum_{n_2 \in \mathcal{V}} Pr(p_2|n_2) = Pr(p_2|v_1) + Pr(p_2|v_2) = 1/3 + 0/3 = 1/3,$$

$$g(m'_{2S}) = C(P, B) = 2/14,$$

$$g(m'_{2V}) = C(P, V) = 4/14.$$

Thus,

$$s(u_1, p_1, m'_2) = 0.4 \cdot (2/3)(2/14)(0) + 0.6 \cdot (2/3)(4/14)(1/3) \approx 0.04$$

and

$$s(u_1, p_2, m'_2) = 0.4 \cdot (2/3)(2/14)(0) + 0.6 \cdot (2/3)(4/14)(1/3) \approx 0.04$$

Hence,

$$a_{u_1, m'_2} = s(u_1, p_1, m'_2) + s(u_1, p_2, m'_2) \approx 0.08. \tag{4.16}$$

Since m'_1 has more weight than m'_2, thus, m'_1 (i.e., items with the same category or the same visual factor) is more important for "User 1" compared to m'_2 (i.e., items with the same brand or the same visual factor).

Example 4.2. (Item-MetaPath Association) Given the same HIN shown in Figure 4.1 and the same probabilistic meta-path $m'_1 = UP\{C \oplus V\}P$ with $\delta = 0.4$, Item-MetaPath association between p_1 and m'_1, a_{p_1,m'_1}, is computed as follows:

$$a_{p_1,m'_1} = \sum_{u \in \mathcal{U}_{p_1}} s(u, p_1, m'_1) = s(u_1, p_1, m'_1) + s(u_2, p_1, m'_1) \tag{4.17}$$

where

$$s(u_1, p_1, m'_1) = \delta \cdot g(u_1, m'_{1S}) g(m'_{1S}) g(m'_{1S}, p_1) + (1 - \delta) \cdot g(u_1, m'_{1V}) g(m'_{1V}) g(m'_{1V}, p_1)$$

and

$$s(u_2, p_1, m'_1) = \delta \cdot g(u_2, m'_{1S}) g(m'_{1S}) g(m'_{1S}, p_1) + (1 - \delta) \cdot g(u_2, m'_{1V}) g(m'_{1V}) g(m'_{1V}, p_1)$$

where

$$\begin{aligned} g(u_1, m'_{1S}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_1) \\ &= Pr(p_1|u_1) + Pr(p_2|u_1) + Pr(p_3|u_1) + Pr(p_4|u_1) + Pr(p_5|u_1) \\ &= 1/3 + 1/3 + 0/3 + 0/3 + 0/3 = 2/3, \end{aligned}$$

$$\begin{aligned} g(u_1, m'_{1V}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_1) \\ &= Pr(p_1|u_1) + Pr(p_2|u_1) + Pr(p_3|u_1) + Pr(p_4|u_1) + Pr(p_5|u_1) \\ &= 1/3 + 1/3 + 0/3 + 0/3 + 0/3 = 2/3, \end{aligned}$$

$$\begin{aligned} g(u_2, m'_{1S}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_2) \\ &= Pr(p_1|u_2) + Pr(p_2|u_2) + Pr(p_3|u_2) + Pr(p_4|u_2) + Pr(p_5|u_2) \\ &= 0/4 + 0/4 + 1/4 + 1/4 + 0/4 = 1/2, \end{aligned}$$

$$\begin{aligned} g(u_2, m'_{1V}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_2) \\ &= Pr(p_1|u_2) + Pr(p_2|u_2) + Pr(p_3|u_2) + Pr(p_4|u_2) + Pr(p_5|u_2) \\ &= 0/4 + 0/4 + 1/4 + 1/4 + 0/4 = 1/2, \end{aligned}$$

$$g(m'_{1S}, p_1) = \sum_{n_2 \in \mathcal{C}} Pr(p_1|n_2) = Pr(p_1|c_1) = 1/3$$

$$g(m'_{1V}, p_1) = \sum_{n_2 \in \mathcal{V}} Pr(p_1|n_2) = Pr(p_1|v_1) + Pr(p_1|v_2) = 1/3 + 0/3 = 1/3,$$

$$g(m'_{1S}) = C(P, C) = 3/14$$

$$g(m'_{1V}) = C(P, V) = 4/14,$$

Thus

$$s(u_1, p_1, m'_1) = 0.4 \cdot (2/3)(3/14)(1/3) + 0.6 \cdot (2/3)(4/14)(1/3) \approx 0.06$$

and

$$s(u_2, p_1, m'_1) = 0.4 \cdot (1/2)(3/14)(1/3) + 0.6 \cdot (1/2)(4/14)(1/3) \approx 0.04$$

Hence,

$$a_{p_1,m'_1} = s(u_1, p_1, m'_1) + s(u_2, p_1, m'_1) \approx 0.1. \tag{4.18}$$

Similarly, for $m'_2 = UP\{B \oplus V\}P$, a_{p_1,m'_2} is calculated as follows:

$$a_{p_1,m'_2} = \sum_{u \in \mathcal{U}_{p_1}} s(u, p_1, m'_2) = s(u_1, p_1, m'_2) + s(u_2, p_1, m'_2)$$

where

$$s(u_1, p_1, m'_2) = \delta \cdot g(u_1, m'_{2S}) g(m'_{2S}) g(m'_{2S}, p_1) + (1 - \delta) \cdot g(u_1, m'_{2V}) g(m'_{2V}) g(m'_{2V}, p_1)$$

and

$$s(u_2, p_1, m'_2) = \delta \cdot g(u_2, m'_{2S}) g(m'_{2S}) g(m'_{2S}, p_1) + (1 - \delta) \cdot g(u_2, m'_{2V}) g(m'_{2V}) g(m'_{2V}, p_1)$$

where

$$\begin{aligned} g(u_1, m'_{2S}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_1) \\ &= Pr(p_1|u_1) + Pr(p_2|u_1) + Pr(p_3|u_1) + Pr(p_4|u_1) + Pr(p_5|u_1) \\ &= 1/3 + 1/3 + 0/3 + 0/3 + 0/3 = 2/3, \end{aligned}$$

$$\begin{aligned} g(u_1, m'_{2V}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_1) \\ &= Pr(p_1|u_1) + Pr(p_2|u_1) + Pr(p_3|u_1) + Pr(p_4|u_1) + Pr(p_5|u_1) \\ &= 1/3 + 1/3 + 0/3 + 0/3 + 0/3 = 2/3, \end{aligned}$$

$$\begin{aligned} g(u_2, m'_{2S}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_2) \\ &= Pr(p_1|u_2) + Pr(p_2|u_2) + Pr(p_3|u_2) + Pr(p_4|u_2) + Pr(p_5|u_2) \\ &= 0/4 + 0/4 + 1/4 + 1/4 + 0/4 = 1/2, \end{aligned}$$

$$\begin{aligned} g(u_2, m'_{2V}) &= \sum_{n_1 \in \mathcal{P}} Pr(n_1|u_2) \\ &= Pr(p_1|u_2) + Pr(p_2|u_2) + Pr(p_3|u_2) + Pr(p_4|u_2) + Pr(p_5|u_2) \\ &= 0/4 + 0/4 + 1/4 + 1/4 + 0/4 = 1/2, \end{aligned}$$

$$g(m'_{2S}, p_1) = \sum_{n_2 \in \mathcal{B}} Pr(p_1|n_2) = Pr(p_1|b_1) = 0,$$

$$g(m'_{2V}, p_1) = \sum_{n_2 \in \mathcal{V}} Pr(p_1|n_2) = Pr(p_1|v_1) + Pr(p_1|v_2) = 1/3 + 0/3 = 1/3,$$

$$g(m'_{2S}) = C(P, B) = 2/14,$$

$$g(m'_{2V}) = C(P, V) = 4/14.$$

Thus,

$$s(u_1, p_1, m'_2) = 0.4 \cdot (2/3)(2/14)(0) + 0.6 \cdot (2/3)(4/14)(1/3) \approx 0.04$$

and

$$s(u_2, p_1, m'_2) = 0.4 \cdot (1/2)(2/14)(0) + 0.6 \cdot (1/2)(4/14)(1/3) \approx 0.03$$

Hence,

$$a_{p_1, m'_2} = s(u_1, p_1, m'_2) + s(u_2, p_1, m'_2) \approx 0.07. \tag{4.19}$$

Since m'_1 has more weight than m'_2, thus, m'_1 (i.e., items with the same category or the same visual factor) is more important for "T-shirt A" compared to m'_2 (i.e., items with the same brand or the same visual factor).

Usually, a group of meta-paths can better explain why a user is interested in an item than a single meta-path. This work uses a group of meta-paths to generate user and item meta-path features. Given a set of meta-paths, multiple User-MetaPath and Item-MetaPath associations can be computed. Such associations of the same user/item can be used to form feature vectors of that user/item. Let $\mathcal{M} = \{m_1, m_2, ..., m_n\}$ be a set of meta-paths where $m_1, m_2, ..., m_n$ are n pre-defined meta-paths. The *user meta-path feature* of u and the *item meta-path feature* of p are defined as

$$\mathbf{f}_u = [a_{u,m_1}, a_{u,m_2}, ..., a_{u,m_n}] \tag{4.20}$$

and

$$\mathbf{f}_p = [a_{p,m_1}, a_{p,m_2}, ..., a_{p,m_n}] \tag{4.21}$$

respectively. Both $\mathbf{f}_u$ and $\mathbf{f}_p$ enclose User-MetaPath and Item-MetaPath associations to represent a given user u and item p. Each dimension in $\mathbf{f}_u$ and $\mathbf{f}_p$ indicates how each meta-path in $\mathcal{M}$ is associated with user u and item p. This can be seen as profiling users/items based on their associations with different meta-paths.

4.2 Meta-Path Based Explainability

In terms of explainability, since each dimension in $\mathbf{f}_u$ and $\mathbf{f}_p$ is meaningful, it can be used to provide explainability in recommendations. For any given user u, item p, and meta-path m, a high value of $a_{u,m}$ indicates that meta-path m potentially connects user u to a substantial number of his/her items. In such instances, it can be inferred that user u expresses a preference for many of their items due to their interconnectedness through meta-path m. Similarly, when $a_{p,m}$ is high, it signifies that meta-path m potentially links item p to a considerable number of its associated users. Consequently, it can be inferred that item p has received interaction from a significant portion of its users, all connected through meta-path m. Therefore, if a recommender system suggests item p to user u, an explanation could be "Because you have shown a preference for numerous items connected to you via m, we recommend item p, which is similarly linked to other users through meta-path m." From this assumption, if both $a_{u,m}$ and $a_{p,m}$ are high, then it can be assumed that m is mutually important for both u and p.

In that case, m is potentially an explanation of why u prefers p, i.e., p is explainable for u based on m. Mathematically, the dot product of $a_{u,m}$ and $a_{p,m}$ can be used to reflect this assumption. Based on this assumption, this work introduces the concept of *meta-path based explainability* for quantifying the explainability between users and items based on multi-hop relations in a HIN.

Definition 4.3. (Meta-Path Based Explainability) *Given a user u, an item p, and a meta-path m, the meta-path based explainability between u and p is measured by the dot product of u's User-MetaPath association and p's Item-MetaPath association.*

From this definition, the higher u and p are associated with the same meta-path m, the higher the explainability between them. The same with existing approaches [72, 203], a threshold value τ can be set. If the computed product is greater than τ, then item p is explainable for user u following meta-path m. Otherwise, item p is not explainable for user u following meta-path m.

Example 4.3. (Meta-Path Based Explainability)

Given the same HIN shown in Figure 4.1 and the same probabilistic meta-paths $m'_1 = UP\{C \oplus V\}P$ and $m'_2 = UP\{B \oplus V\}P$, a_{p_1,m'_2} with $\delta = 0.4$, from Examples 4.1 and 4.2, $a_{u_1,m'_1} \approx 0.12$, $a_{u_1,m'_2} \approx 0.08$, $a_{p_1,m'_1} \approx 0.1$, and $a_{p_1,m'_2} \approx 0.07$.

Following a similar calculation process, the Item-MetaPath association between p_3 and m'_1 and the Item-MetaPath association between p_3 and m'_2 are $a_{p_3,m'_1} \approx 0.03$ and $a_{p_3,m'_2} \approx 0.04$, respectively.

Thus, the user meta-path feature of u_1 is

$$\mathbf{f}_{u_1} = [a_{u_1,m'_1}, a_{u_1,m'_2}] = [0.12, 0.08] \tag{4.22}$$

Meanwhile, the item meta-path feature of p_1 is

$$\mathbf{f}_{p_1} = [a_{p_1,m'_1}, a_{p_1,m'_2}] = [0.1, 0.07] \tag{4.23}$$

Similarly,, the item meta-path feature of p_4 is

$$\mathbf{f}_{p_4} = [a_{p_4,m'_1}, a_{p_4,m'_2}] = [0.03, 0.04] \tag{4.24}$$

By computing the dot product of $\mathbf{f}_{u_1}$ and $\mathbf{f}_{p_1}$, the meta-path based explainability between u_1 and p_1 is

$$\mathbf{f}_{u_1} \cdot \mathbf{f}_{p_1} = [0.12, 0.08] \cdot [0.1, 0.07] = 0.0176 \tag{4.25}$$

Similarly, the meta-path based explainability between u_1 and p_4 is

$$\mathbf{f}_{u_1} \cdot \mathbf{f}_{p_4} = [0.12, 0.08] \cdot [0.03, 0.04] = 0.0068 \tag{4.26}$$

Since $\mathbf{f}_{u_1} \cdot \mathbf{f}_{p_1} > \mathbf{f}_{u_1} \cdot \mathbf{f}_{p_4}$, p_1 is more explainable to u_1 than p_4.

4.3 The Proposed Explainable Recommendation Method

In this section, how to generate explainable recommendations based on the meta-path features is described. The proposed framework is based on a Bayesian Personalized Ranking Matrix Factorization (BPR-MF) framework. It is modified to integrate the meta-path based explainability into the learning framework. The traditional BPR-MF model ranks the candidate items based on the user-personalized recommendation scores. The recommendation score of a user u towards an item p denoted as $\hat{x}_{up}$ is computed by

$$\hat{x}_{up} = \alpha + \beta_u + \beta_p + \boldsymbol{\gamma_u}^T \boldsymbol{\gamma_p} \tag{4.27}$$

where α is a global offset, β_u and β_p are user and item bias terms, $\boldsymbol{\gamma_u}$ and $\boldsymbol{\gamma_p}$ are K_1-dimensional vectors of user u and item p latent factors respectively. The system is learned by using positive and negative items in a dataset. For any user $u \in \mathcal{U}$, let $\mathcal{P}_u^+$ be a set of positive items of user u. A training sample set is defined as

$$\mathcal{D}_S = \{(u, p, q) | u \in \mathcal{U} \wedge p \in \mathcal{P}_u^+ \wedge q \in \mathcal{P} \setminus \mathcal{P}_u^+\} \tag{4.28}$$

where p is a user's positive item and q is a user's negative item which is an unobserved item of a user u. A stochastic gradient-descent algorithm is adopted for training with a generic optimization criterion defined as follows:

$$\sum_{(u,p,q) \in \mathcal{D}_S} -ln\ \sigma(\hat{x}_{up} - \hat{x}_{uq}) + \lambda_\Theta ||\Theta||^2 \tag{4.29}$$

where $\hat{x}_{up}$ and $\hat{x}_{uq}$ are the recommendation scores of user u towards p and q respectively, σ is the sigmoid function and $||\Theta||^2$ is an L2 norm regularization term where λ_Θ is a regularization hyper-parameter and Θ denotes model parameters. The purpose is to differentiate a pair of user's positive and user's negative items by minimizing the loss in Eq. (4.29).

The traditional BPR-MF model involves only user-item interaction data for learning. In [37], the BPR-MF model was extended to incorporate visual information from item images. User and item latent visual factors were introduced to the traditional model. For each item,

item latent visual factors are computed by projecting its image feature onto the visual latent space. This projection is done by using a projection matrix. This matrix is learned during model training along with the traditional user/item latent factors. Meanwhile, since there are no images of users, user latent visual factors in visual rating space are directly learned without projecting as item latent visual factors. Following this idea, the meta-path based features of both u and p can be integrated into the personalized recommendation score as follows:

$$\hat{x}_{up} = \alpha + \beta_u + \beta_p + \boldsymbol{\gamma_u}^T \boldsymbol{\gamma_p} + \boldsymbol{\theta_u}^T \boldsymbol{\theta_p} + \boldsymbol{\beta}_P^T \mathbf{f}_p + \boldsymbol{\beta}_U^T \mathbf{f}_u \tag{4.30}$$

where $\boldsymbol{\theta_u}$ and $\boldsymbol{\theta_p}$ are additional K_2-dimensional latent factors apart from the traditional latent factors $\boldsymbol{\gamma_u}$ and $\boldsymbol{\gamma_p}$, $\boldsymbol{\beta}_P$ is an item feature bias vector, and $\boldsymbol{\beta}_U$ is a user feature bias vector. These additional latent factors are called meta-path based latent factors since they are factorized based on the proposed user/item meta-path based features. They are computed by

$$\boldsymbol{\theta_u} = \mathbf{E}_U \mathbf{f}_u \tag{4.31}$$

and

$$\boldsymbol{\theta_p} = \mathbf{E}_P \mathbf{f}_p \tag{4.32}$$

where $\mathbf{E}_U$ and $\mathbf{E}_P$ are matrices projecting $\mathbf{f}_u$ and $\mathbf{f}_p$ into K_2-dimensional latent spaces respectively. Both $\mathbf{E}_U$ and $\mathbf{E}_P$ are additional parameters in this model. Overall, $\hat{x}_{up}$ is calculated from two parts, the traditional latent factors $\boldsymbol{\gamma_u}$ and $\boldsymbol{\gamma_p}$ (including their biases α, β_u and β_p) and the meta-path based latent factors $\boldsymbol{\theta_u}$ and $\boldsymbol{\theta_p}$ (including their biases $\boldsymbol{\beta}_P^T \mathbf{f}_p$ and $\boldsymbol{\beta}_U^T \mathbf{f}_u$). Unlike VBPR, the proposed model also considers the feature from the user side to learn the additional latent factors of a user. Compared to most HIN-based models, it is also worth noting that the proposed model can be used to incorporate multi-hop information from a set of meta-paths. In other words, instead of relying on a single meta-path, multiple meta-paths can be leveraged altogether simultaneously. Also, any combination of meta-paths can be applied in this approach. This includes a combination of regular meta-paths, probabilistic meta-paths, and both.

Next, how to utilize the meta-path based features to increase the explainability of the proposed model is described. In [203], an explainable MF model which is a modification of the traditional MF model was proposed. This model jointly considers user-item interactions as in the traditional MF model and the explainability scores of user-item pairs as an additional soft constraint in the loss function. To measure the explainability between u and p based on a set of meta-paths $\mathcal{M}$, the explainability score E_{up} can be computed by the dot product of $\mathbf{f}_u$

and $\mathbf{f}_p$. Since $\mathbf{f}_u$ and $\mathbf{f}_p$ are two vectors and can have significantly different vector magnitudes, the cosine similarity which is the normalized dot product of two vectors is used to compute E_{up} as:

$$E_{up} = h(\mathbf{f}_u, \mathbf{f}_p) \tag{4.33}$$

where $h(\mathbf{f}_u, \mathbf{f}_p)$ denotes the cosine similarity between $\mathbf{f}_u$ and $\mathbf{f}_p$. Note that, alternative similarity functions can be considered. Cosine similarity was chosen for its computational simplicity and widespread use in measuring vector similarity. Based on Definition (4.3), if p (or q) is explainable for u, then they should be close to each other in the latent space. Based on this assumption, the explainability scores are integrated into the loss function to constrain the distance between the user and item latent factors. The higher the explainability score, the closer both latent factors are. Thus, the original loss function in Eq. (4.29) is changed to

$$\sum_{(u,p,q)\in\mathcal{D}_S} -ln\ \sigma(\hat{x}_{up} - \hat{x}_{uq}) + \frac{\lambda_\Theta}{2}||\Theta||^2 + \frac{\lambda_E}{2}\left(||\mathbf{u}-\mathbf{p}||^2 E_{up} + ||\mathbf{u}-\mathbf{q}||^2 E_{uq}\right) \tag{4.34}$$

where $\mathbf{u} = [\boldsymbol{\gamma_u}; \boldsymbol{\theta_u}]$, $\mathbf{p} = [\boldsymbol{\gamma_p}; \boldsymbol{\theta_p}]$ and $\mathbf{q} = [\boldsymbol{\gamma_q}; \boldsymbol{\theta_q}]$ denote the final combined latent factors of u, p and q respectively, E_{up} and E_{uq} are the explainability scores and λ_E is a regularization hyper-parameter. If E_{up} is high, it will constrain $||\mathbf{u}-\mathbf{p}||$ to be lower to minimize the loss. Thus, $\mathbf{u}$ and $\mathbf{p}$ will be closer in the latent space. The same process applies for E_{uq} and the distance between $\mathbf{u}$ and $\mathbf{q}$. In this way, the meta-path features are used to constrain the recommender system to make recommendations with high meta-path based explainability instead of any recommendations. Thus, given a set of meta-paths used for feature extraction, the recommendations made based on the extracted features can be explained by the meanings of these meta-paths. The proposed framework utilizing the meta-path features and the meta-path based explainability scores is illustrated in Figure 4.4. It is worth noting that the proposed framework uses a set of pre-defined meta-paths to constrain the explainability of recommendations. This is different from the previous work attempting to extract explanations along with predictions. For instance, in [205], meta-paths were not used during the learning process but were extracted as explanations along with the outputs. Also, compared to existing studies on using pre-defined meta-paths to improve explainability, the proposed framework addresses the issue of scalability and is more flexible. For example, compared to [206], meta-paths used in the proposed framework are not limited to only symmetric meta-paths of length 3.

Complexity Analysis In the proposed approach, the meta-path features $\mathbf{f}_u$ and $\mathbf{f}_p$ are computed as part of pre-processing. Given a meta-path $m = UN_1N_2\cdots N_lP$, let n be the

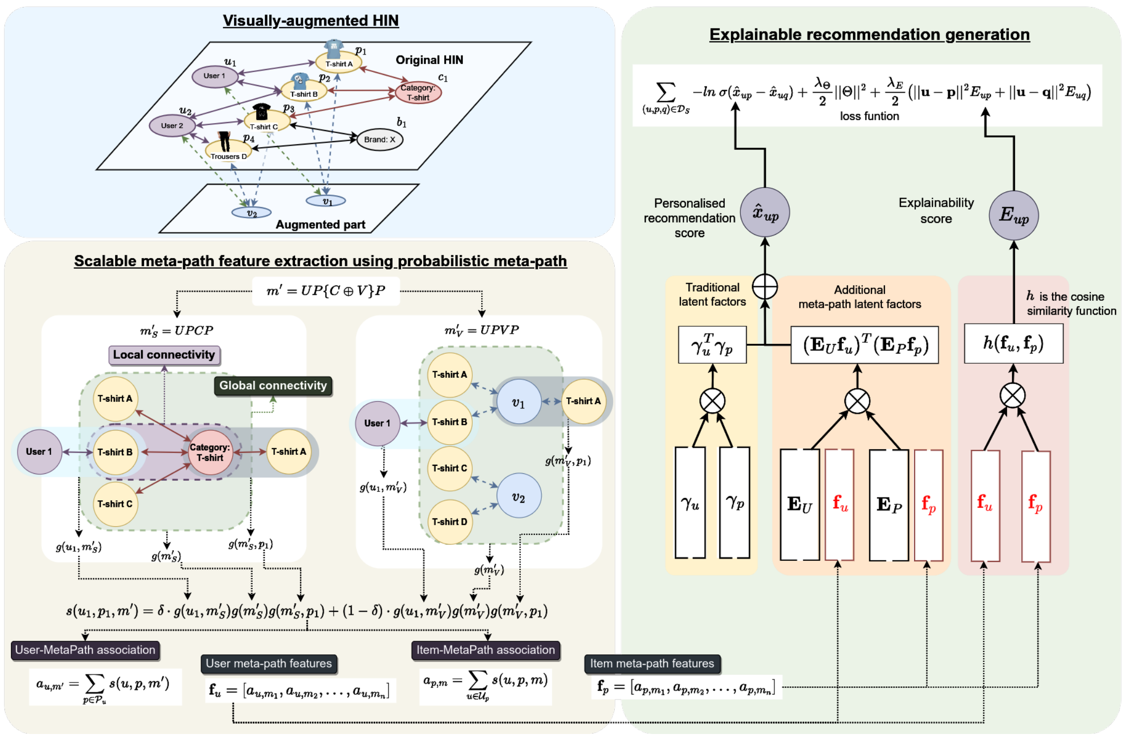

Visually-augmented HIN
Original HIN
Augmented part
User 1
User 2
T-shirt A
T-shirt B
T-shirt C
Trousers D
Category: T-shirt
Brand: X
Scalable meta-path feature extraction using probabilistic meta-path
$m' = UP\{C \oplus V\}P$
$m'_S = UPCP$
$m'_V = UPVP$
Local connectivity
Global connectivity
$s(u_1, p_1, m') = \delta \cdot g(u_1, m'_S) g(m'_S) g(m'_S, p_1) + (1-\delta) \cdot g(u_1, m'_V) g(m'_V) g(m'_V, p_1)$
User-MetaPath association
$a_{u,m'} = \sum_{p \in \mathcal{P}_u} s(u,p,m')$
User meta-path features
$\mathbf{f}_u = [a_{u,m_1}, a_{u,m_2}, \ldots, a_{u,m_n}]$
Item-MetaPath association
$a_{p,m} = \sum_{u \in \mathcal{U}_p} s(u,p,m)$
Item meta-path features
$\mathbf{f}_p = [a_{p,m_1}, a_{p,m_2}, \ldots, a_{p,m_n}]$
Explainable recommendation generation
$\sum_{(u,p,q) \in \mathcal{D}_S} -ln\, \sigma(\hat{x}_{up} - \hat{x}_{uq}) + \frac{\lambda_\Theta}{2}||\Theta||^2 + \frac{\lambda_E}{2}(||\mathbf{u}-\mathbf{p}||^2 E_{up} + ||\mathbf{u}-\mathbf{q}||^2 E_{uq})$
loss funtion
Personalised recommendation score
Explainability score
Traditional latent factors
Additional meta-path latent factors
h is the cosine similarity function
$\gamma_u^T \gamma_p$
$(\mathbf{E}_U \mathbf{f}_u)^T (\mathbf{E}_P \mathbf{f}_p)$
$h(\mathbf{f}_u, \mathbf{f}_p)$

Figure 4.4: The overall framework of SEV-RS which consists of three parts: (1) a visually-augmented HIN, containing visual information extracted from item images (see Chapter 3), (2) scalable meta-path feature extraction, extracting user and item meta-path features for learning recommendations and computing meta-path based explainability scores of user-item pairs., and (3) explainable recommendation method, a modified BPR-MF model considering the extracted user and item meta-path features along with meta-path based explainability scores for making visually-aware recommendations.

average number of adjacent nodes per node. Computing $s(u, p, m)$ by considering all possible path instances requires $O(n^l)$. This is more computationally expensive than the proposed method. From Eq. (4.7), computing $g(m)$ needs $O((l-1)n^2)$. Meanwhile, computing $\sum_{n_1 \in \mathcal{N}_1} Pr(n_1|u)$ and $\sum_{n_l \in \mathcal{N}_l} Pr(p|n_l)$ requires $O(n)$. In total, for any pair of a user/item and a meta-path, computing $s(u, p, m)$ requires $O((l-1)n^2) + O(n)$. Furthermore, $g(m)$ only depends on a meta-path. It can be pre-calculated once and used for all users/items. As a result, the proposed method is more scalable compared to the method that uses actual path instances.

As for explainable recommendation generation, the modified BPR-MF framework consists of the traditional part and the additional part as previously discussed. The first part requires $O(K_1)$ to update the user and item latent factors for each iteration. For the additional part, updating $\mathbf{E}_U$ and $\mathbf{E}_P$ needs $O(K_2|\mathcal{M}|)$. Updating $\boldsymbol{\beta}_U$ and $\boldsymbol{\beta}_P$ needs $O(K_2)$. Therefore, the proposed learning framework requires $O(K_2|\mathcal{M}|) + O(K_2)$, in addition to the traditional part of the BPR-MF model. This is scalable since the size of meta-path set $|\mathcal{M}|$ and the sizes of latent factors K_1 and K_2 are usually small.

4.4 Experimental Setup

In this section, the details of the experiments are provided. The experiments were conducted aiming to answer the following questions: How does the proposed approach using the meta-path features perform compared with the baselines?, How does the proposed approach perform when it is applied to a visually-augmented HIN compared with the baselines?, How does the proposed approach perform when the meta-path based explainability is included compared with the baselines?, and Is the proposed approach scalable compared with the baselines?

4.4.1 Datasets

The experiments were conducted on two real-world datasets, i.e., **MovieLens dataset** and **Amazon dataset**, which were also used in the previous chapter (See Section 3.5.1). In these experiments, more types of nodes and relations were considered. For **MovieLens dataset**, 7 node types and 14 relation types (inverse relation types included) were considered. For **Amazon** dataset, 6 node types and 12 relation types (inverse relation types included) were considered. The lists of node and relations types in both datasets are shown in Table 4.1.

For both datasets, those users who have less than two items and those items that have been interacted with by less than two users were filtered out. The visually-augmented HINs were constructed as described in Section 3.2. Only one visual factor type was considered which is CNN visual factors. It was selected based on its general performance from the results of the experiments in Chapter 3. It should be noted that more types of nodes and relations were considered in the experiments in this chapter compared to the experiments in Chapter 3. 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). These node and relation types as well as basic statistics of the visually-augmented HINs of both datasets are shown in Table 4.1.

Table 4.1: 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

4.4.2 Parameterization and Experiment Environment

The number of visual factors k_V is 100 (i.e., $k = 100$ in the k-means clustering method). The number of representative visual factors per user is 1 ($k_s = 1$). The meta-paths used for generating the meta-path based features for both datasets are selected from the literature [122, 173]. They are shown in Table 4.2 where the second column lists the regular meta-paths while the third column lists the probabilistic meta-paths. The sizes of user/item latent factors, K_1 and K_2, were set to 150. Therefore, the final latent factors, $\mathbf{u}$ and $\mathbf{p}$, are 300-dimensional. $\lambda_\Theta = 5 \times 10^{-5}$ was applied for both datasets. All experiments were conducted on a machine with dual-core Intel(R) 1.80GHz CPU, NVIDIA 16GB GPU, and 128GB RAM.

4.4.3 Evaluation Metrics

The proposed approach was evaluated in the Top-N recommendation task. Three evaluation aspects, i.e., Accuracy, Explainability, and Scalability were considered. As for Accuracy, it was evaluated by three commonly used metrics: *Mean Average Precision@N* (MAP@N), *Mean Recall@N* (Recall@N), and *Mean F1 Score@N* (F1@N) with $N = 1, 5, 10, 50, 100$. MAP@N is

Table 4.2: The meta-paths used in the experiments on MovieLens and Amazon datasets

Dataset	Meta-paths		Probabilistic meta-paths	
MovieLens	$UPUP$,	$UPUPUP$,	$UP\{U \oplus V\}P$,	$UP\{U \oplus V\}P\{U \oplus V\}$,
	$UPGP$,	$UPGPUP$,	$UP\{G \oplus V\}P$,	$UP\{G \oplus V\}P\{U \oplus V\}$,
	$UPAP$,	$UPAPUP$,	$UP\{A \oplus V\}P$,	$UP\{A \oplus V\}P\{U \oplus V\}$,
	$UPDP$,	$UPDPUP$,	$UP\{D \oplus V\}P$,	$UP\{D \oplus V\}P\{U \oplus V\}$,
	$UPTP$,	$UPTPTP$	$UP\{T \oplus V\}P$,	$UP\{T \oplus V\}P\{T \oplus V\}$
Amazon	$UPUP$,	$UPUPUP$,	$UP\{U \oplus V\}P$,	$UP\{U \oplus V\}P\{U \oplus V\}$,
	$UPCP$,	$UPCPUP$,	$UP\{C \oplus V\}P$,	$UP\{C \oplus V\}P\{U \oplus V\}$,
	$UPBP$,	$UPBPUP$,	$UP\{B \oplus V\}P$,	$UP\{B \oplus V\}P\{U \oplus V\}$,
	$UPHP$,	$UPHPHP$	$UP\{H \oplus V\}P$,	$UP\{H \oplus V\}P\{H \oplus V\}$

computed as follows:

$$MAP@N = \frac{1}{|\mathcal{U}|}\sum_{u \in \mathcal{U}} (AP@N)_u \tag{4.35}$$

where $(AP@N)_u$ is the average precision@N of user u computed by

$$(AP@N)_u = \frac{1}{min(|\mathcal{P}_u|, N)} \sum_{k=1}^{N} \frac{|\mathcal{P}_u \cap \mathcal{P}_u^k|}{k} \cdot rel_u(k) \tag{4.36}$$

where $\mathcal{U}$ is a set of users, $\mathcal{P}_u$ is the set of user u's items, $\mathcal{P}_u^k$ is the set of top-k recommended items of user u, and $rel_u(k)$ is an indicator function indicating whether the kth item in the top-N recommendation list of user u was relevant (i.e., $rel_u(k) = 1$ if this kth item is correct; otherwise, $rel_u(k) = 0$).

To evaluate Explainability, three metrics were adopted, i.e., *Mean Explainability Precision@N* (EP@N) [203], *Mean Explainability Recall@N* (ER@N) [203], and *Mean Explainability F1 Score@N* (EF@N) [90] defined as follows:

$$EP@N = \frac{1}{|\mathcal{U}|}\sum_{u \in \mathcal{U}} \frac{|\mathcal{E}_u \bigcap \mathcal{Y}_u|}{|\mathcal{Y}_u|}, \tag{4.37}$$

$$ER@N = \frac{1}{|\mathcal{U}|}\sum_{u \in \mathcal{U}} \frac{|\mathcal{E}_u \bigcap \mathcal{Y}_u|}{|\mathcal{E}_u|}, \tag{4.38}$$

and

$$EF@N = 2 \cdot \frac{EP@N \cdot ER@N}{EP@N + ER@N} \tag{4.39}$$

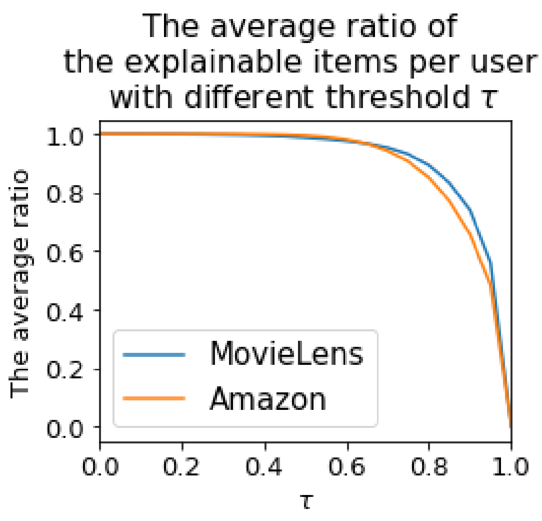


Figure 4.5: The average ratio of the explainable items per user with different threshold value τ to validate the explainable items of each user

where $\mathcal{U}$ denotes the users set, $\mathcal{E}_u$ denotes the set of explainable items of user u and $\mathcal{Y}_u$ denotes the set of Top-N recommended items of user u. For each user, the explainable items of that user are determined as in Definition 4.3, given the set of meta-paths defined in Table 4.2. Similarly to [72, 203], one can set up a threshold value τ to validate the explainable items of each user. Specifically, p is explainable for u if $h(\mathbf{f}_u, \mathbf{f}_p) \geq \tau$ where τ is a pre-defined threshold. Figure 4.5 shows the average ratio of the explainable items to the user's items of each user in both datasets when τ is varied from 0 to 1. The ratio decreases as τ increases. To include most of the explainable items, $\tau = 0.55$ was applied for both MovieLens and Amazon datasets for evaluation. $EP@5$ and $ER@5$ were selected to evaluate the explainability performance.

4.5 Results and Discussions

4.5.1 Comparison of the proposed approach and the baseline

To examine the performance of the proposed approach using the meta-path features without visual information involved, two variations of the proposed approach applied on regular HINs were compared with the baselines as follows:

- **CF**: the CF-KNN model [14] that uses only user-item interactions.
- **BPR** [79]: the traditional BPR-MF model using user-item interactions.

- **VBPR** [37]: the modified BPR-MF model that jointly leverages user-item interactions and visual information. The same CNN features used in the proposed approach were used as visual information in this model.
- **DVBPR** [38]: the modified version of VBPR that jointly trains the visual feature extraction model with the recommendation model instead of using the features from the pre-trained model.
- **DeepStyle** [80]: the BPR-MF model that incorporates the style features of items computed by subtracting item category representations from the visual features generated by CNN. The same CNN features were used as in VBPR for computing these style features.
- **MV** [173]: an approach using metapath2vec [85] with the CF-KNN model. Multiple models of this approach were built based on each of the meta-paths in Table 4.2. The best model was selected for comparison.
- **GA** [5]: the state-of-the-art HIN-based model using Graph Attention Network. This model was applied to HINs without visual information.
- **GA-v** [5]: the **GA** approach applied to visually-augmented HINs.
- **PM**: the proposed model that uses regular meta-paths with regular HINs. Visual information and the meta-path based explainability were not considered.
- **PM-v**: the proposed model that uses probabilistic meta-paths with visually-augmented HINs. The meta-path explainability was not considered. The parameter δ was varied among $\{0, 0.1, 0.2, ..., 1\}$ and the result with $\delta = 0.2$ were selected for comparison for both datasets.

For fair comparisons, the size of the final user/item latent factors or embeddings in **BPR**, **VBPR**, **DVBPR**, **DeepStyle**, **MV**, **GA** and **GA-v** were identically set to 300 as in the proposed models. For **CF** and **MV**, the size of neighborhoods was set to 10. Other hyperparameter settings for the baselines were set as in their papers.

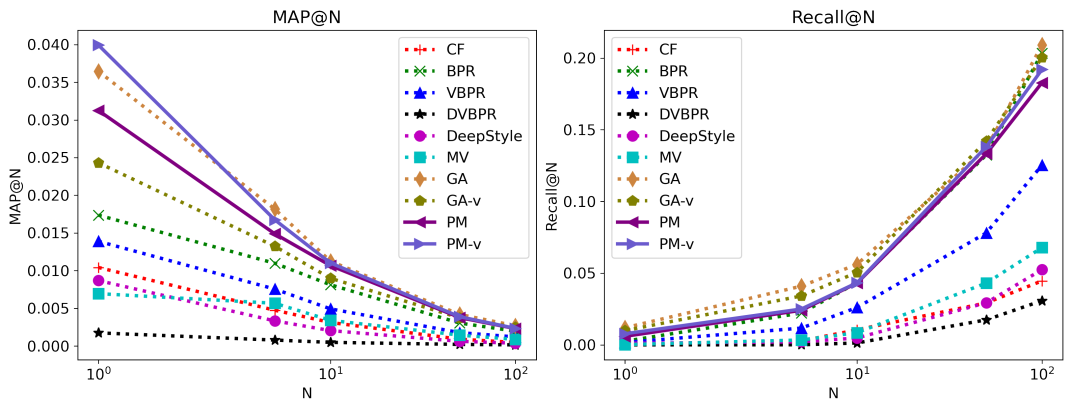
The F1@N results of **PM** and **PM-v**, compared with the baselines on both **MovieLens** and **Amazon** datasets, are presented in Table 4.3. In this table, the highest values in each column are highlighted in bold, and the second-highest values are underlined. According to the results, on **MovieLens** dataset, **GA** outperformed other models. **GA-v** achieved the

Table 4.3: F1@N results of the baselines and the variations of the proposed approach (without the explainability component), **PM** and **PM-v**, on **MovieLens** and **Amazon** datasets. The highest values in each column are highlighted in bold, while the second-highest values are underlined.

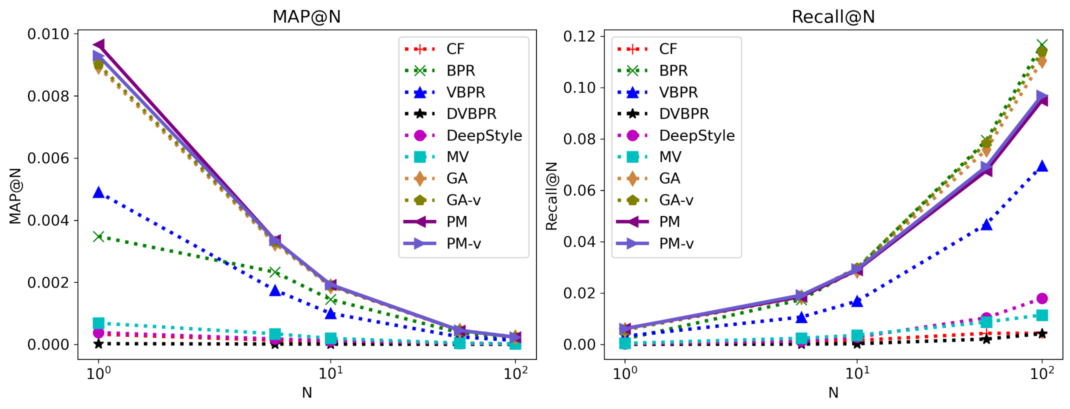
Model	MovieLens					Amazon				
	F1@1	F1@5	F1@10	F1@50	F1@100	F1@1	F1@5	F1@10	F1@50	F1@100
CF	0.0006	0.0000	0.0048	0.0017	0.0010	0.00025	0.00023	0.00015	0.00004	0.00002
BPR	0.0045	0.0147	0.0136	0.0059	0.0040	0.00264	0.00411	0.00277	0.00075	0.00040
VBPR	0.0030	0.0092	0.0082	0.0035	0.0024	0.00390	0.00301	0.00189	0.00049	0.00026
DVBPR	0.0000	0.0002	0.0007	0.0004	0.0002	0.00003	0.00002	0.00002	0.00001	0.00001
DeepStyle	0.0020	0.0026	0.0028	0.0012	0.0008	0.00029	0.00031	0.00023	0.00007	0.00004
MV	0.0004	0.0043	0.0049	0.0027	0.0018	0.00055	0.00060	0.00037	0.00009	0.00005
GA	**0.0181**	**0.0252**	**0.0187**	**0.0082**	**0.0053**	0.00688	0.00554	0.00351	0.00089	0.00047
GA-v	0.0142	0.0190	0.0153	0.0072	0.0049	0.00702	0.00562	0.00357	**0.00091**	**0.00048**
PM	0.0100	0.0184	0.0171	0.0073	0.0045	**0.00753**	0.00570	0.00360	0.00089	0.00046
PM-v	0.0130	0.0201	0.0176	0.0075	0.0046	0.00736	**0.00572**	**0.00362**	0.00090	0.00047

second-best performance when $N = 1$ and 100, while the proposed model **PM-v**, utilizing probabilistic meta-paths with visually-augmented HINs, achieved the second-best performance for $N = 5, 10,$ and 50. These results suggest that, although the proposed model **PM-v** surpassed most baselines, its performance was slightly lower than that of **GA** and **GA-v**. However, Section 4.5.3 will illustrate that the proposed approach is significantly more scalable than this deep-learning-based approach while achieving comparable performance. On **Amazon** dataset, **PM** and **PM-v** outperformed the other baselines for $N = 1, 5$ and 10. Specifically, **PM** achieved the best performance when $N = 1$ while **PM-v** achieved the best performance when $N = 5$ and 10. For $N = 50$ and 100, **GA-v** attained the best performance while **PM-v** attained the second-best performance. These results indicate the effectiveness of the proposed models on this dataset, as they consistently outperformed the baselines across various values of N and demonstrated comparable performances when compared to the top-performing baselines, **GA** and **GA-v**.

Figure 4.6a and Figure 4.6b show the MAP@N and Recall@N results on **MovieLens** dataset and the results on **Amazon** dataset respectively. First, the performance of **PM** is discussed in comparison with the other baselines. From these figures, **PM** outperformed **CF** in terms of both MAP@N and Recall@N on both datasets. This can be explained that **CF** only uses single-hop relations (user-item interactions) for learning while the multi-hop relations are ignored in



(a) MovieLens



(b) Amazon

Figure 4.6: Accuracy (MAP@N and Recall@N) comparison between the proposed approaches (without the explainability component), **PM** and **PM-v**, and other baselines on (a) **MovieLens** and (b) **Amazon** datasets.

the model. **PM** outperformed **BPR** in terms of MAP@N but performed similarly to **BPR** in terms of Recall@N on both datasets. Although **BPR** uses matrix factorization and negative sampling to overcome the sparsity problem, it still relies on user-item interactions to learn users' preferences. The proposed model **PM** involves high-order information obtained from HINs in addition to user-item historical data. From the result, the improvement of MAP@N indicates the effectiveness of leveraging such information in a BPR-MF model. Compared with the visually-aware baseline models including **VBPR**, **DVBPR** and **DeepStyle**, **PM** performed better than all of these models in terms of MAP@N and Recall@N on both datasets. This suggests that, compared to those models using visual features as side information, the proposed model using multi-hop relations without visual information is more effective. Therefore, between using only visual information extracted from item images and using multi-hop relations in a HIN without visual information, the latter may allow for better performance in terms of Accuracy. **PM** also outperformed **MV** in terms of both MAP@N and Recall@N on both datasets. Although **MV** also utilizes meta-path based multi-hop relations, it can only consider a single meta-path at a time. On the other hand, the proposed approach can consider multiple meta-paths simultaneously. This provides an advantage for the proposed model **PM**. The deep-learning based model **GA** outperformed **PM** on **MovieLens** dataset but they both performed similarly on **Amazon** dataset. This shows that, with only semantic information in a network, **GA** can produce more accurate recommendations than the proposed model. This indicates that the proposed model **PM** is not as effective as the state-of-the-art deep learning model when visual information is not considered.

Next, how **PM-v** performed compared with the other models is discussed. The results of **PM-v** are similar to the results of **PM** when compared to the baselines using only user-item interaction information, i.e., **CF** and **BPR**. As shown in Figure 4.6, similar to **PM**, **PM-v** outperformed **CF** in terms of both MAP@N and Recall@N on both datasets. This emphasizes the effectiveness of using multi-hop relations to improve Accuracy. **PM-v** also outperformed **BPR** in terms of MAP@N while performed similarly to it in terms of Recall@N on both datasets. The reason could be the same as in the case of **PM**. This demonstrates the effectiveness of including multi-hop relations to improve Accuracy as evidenced by the higher MAP@N of **PM-v**. Compared to those baselines that can utilize visual information, **PM-v** performed better than **VBPR**, **DVBPR**, and **DeepStyle** in terms of both Precision and Recall on both **MovieLens** and **Amazon** dataset. This shows that the proposed model leveraged visual information to produce accurate recommendations more effectively than these visually-aware BPR-based models. Compared to **MV**, similar to **PM**, **PM-v** also outperformed **MV** in terms of both metrics on both datasets. This demonstrates the effectiveness of using multiple meta-paths including probabilistic meta-paths as opposed to using a single meta-path to leverage multi-hop relations in HINs. Comparing the two variations of the proposed approach, **PM-v** performed better than **PM** on **MovieLens** dataset. This suggests that the performance of the proposed approach increased when using the visually-augmented HIN on this dataset. In fact, the performance of **PM-v** was enhanced up to the performance of **GA** which is a deep learning model. Also, it should be noted that **GA-v** performed worse than **GA** in terms of both MAP@N and Recall@N on **MovieLens** dataset. This implies that the performance of the Graph Attention model dropped when it is applied to the visually-augmented HIN on this dataset. This result suggests that the Graph Attention model may not work well on the augmented HIN unlike the proposed approach **PM-v**. This demonstrates the effectiveness of the proposed approach in leveraging visual information from a visually-augmented HIN. On **Amazon** dataset, **GA**, **GA-v**, **PM**, and **PM-v** all performed similarly in terms of MAP@N as shown in Figure 4.6b. In terms of Recall@N, **PM** and **PM-v** performed slightly worse than **GA** and **GA-v**. One possible reason is that **Amazon** dataset contains numerous cold-start users which may limit the performances of these models.

4.5.2 Explainability Discussion

In this part, the proposed approach involving the meta-path based explainability is evaluated. The same baselines as in the previous experiment were compared with two variations of the proposed approach. These two variations are as follows:

- **xPM**: the proposed explainable model using regular meta-paths with regular HINs. This variation is used to examine how the meta-path based explainability increases the recommendation explainability.
- **xPM-v**: the proposed explainable model using probabilistic meta-paths with visually-augmented HINs. This variation is used to examine its effectiveness when both visual information and explainability are considered.

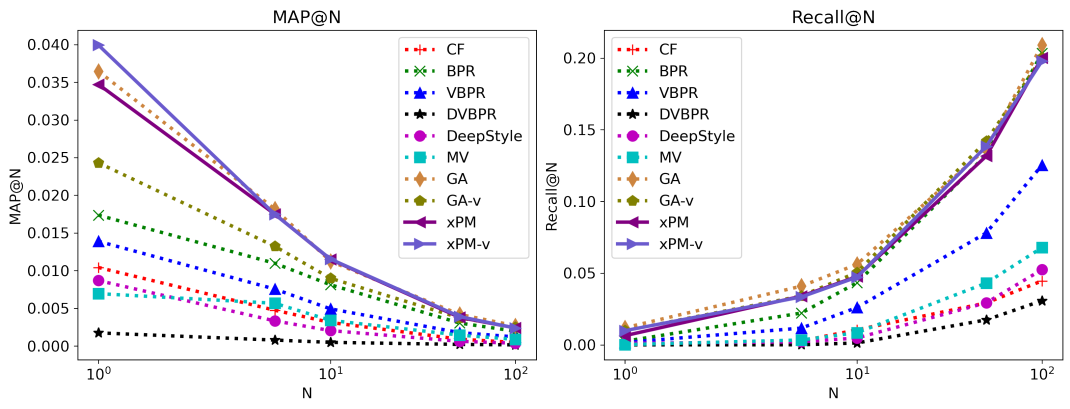
First, the Accuracy performance of the proposed explainable approaches is discussed. Table 4.4 shows the F1@N results of **xPM** and **xPM-v**, compared with the baselines on both **MovieLens** and **Amazon** datasets. In this table, the highest values in each column are highlighted in bold, and the second-highest values are underlined. The results on **MovieLens** dataset demonstrate that **GA** achieved the best performance compared to the others. The proposed model **xPM** performed as the second-best when $N = 5$ and 10, **xPM-v** attained the second-best performance for $N = 1, 10$, and 50, and **GA-v** attained the second-best performance when $N = 100$. These results suggest that, on **MovieLens** dataset, both proposed models **xPM** and **xPM-v** outperformed almost every baseline except **GA**. However, similar to the results of **PM** and **PM-v**, the scalability performances of **xPM** and **xPM-v** are significantly better than **GA**, as will be shown in the following section. On the other hand, on **Amazon** dataset, the proposed model **xPM-v** demonstrated the best performance for every N compared to the other baselines including **GA** and **GA-v**. Also, **xPM** performed as the second best for every N. These results illustrate the effectiveness of incorporating the visual information and the meta-path based explainability in the proposed approach.

Table 4.4: F1@N results of the baselines and the variations of the proposed approach (with the explainability component), **xPM** and **xPM-v**, on **MovieLens** and **Amazon** datasets. The highest values in each column are highlighted in bold, while the second-highest values are underlined.

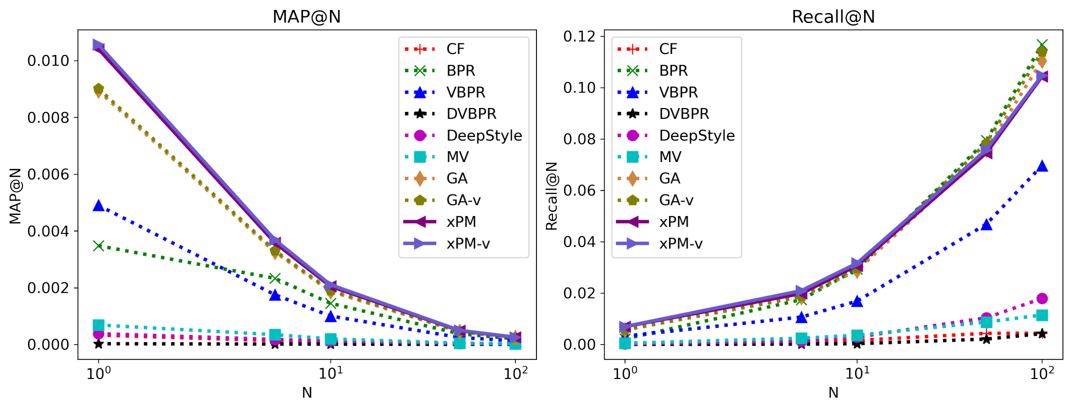
Model	MovieLens					Amazon				
	F1@1	F1@5	F1@10	F1@50	F1@100	F1@1	F1@5	F1@10	F1@50	F1@100
CF	0.0006	0.0000	0.0048	0.0017	0.0010	0.00025	0.00023	0.00015	0.00004	0.00002
BPR	0.0045	0.0147	0.0136	0.0059	0.0040	0.00264	0.00411	0.00277	0.00075	0.00040
VBPR	0.0030	0.0092	0.0082	0.0035	0.0024	0.00390	0.00301	0.00189	0.00049	0.00026
DVBPR	0.0000	0.0002	0.0007	0.0004	0.0002	0.00003	0.00002	0.00002	0.00001	0.00001
DeepStyle	0.0020	0.0026	0.0028	0.0012	0.0008	0.00029	0.00031	0.00023	0.00007	0.00004
MV	0.0004	0.0043	0.0049	0.0027	0.0018	0.00055	0.00060	0.00037	0.00009	0.00005
GA	**0.0181**	**0.0252**	**0.0187**	**0.0082**	**0.0053**	0.00688	0.00554	0.00351	0.00089	0.00047
GA-V	0.0142	0.0190	0.0153	0.0072	0.0049	0.00702	0.00562	0.00357	0.00091	0.00048
xPM	0.0109	0.0231	0.0185	0.0074	0.0047	0.00831	0.00607	0.00384	0.00096	0.00050
xPM-v	0.0159	0.0230	0.0185	0.0076	0.0047	**0.00840**	**0.00628**	**0.00395**	**0.00098**	**0.00051**

Considering the MAP@N and Recall@N results on **MovieLens** dataset in Figure 4.7a, **xPM** outperformed almost every baseline except **GA** in terms of MAP@N. In terms of Recall@N, **xPM** performed similarly to **GA** and **GA-v** while outperforming the others. This suggests that, despite the explainability component included in this model, **xPM** can still produce recommendations as effectively as **PM**. Similar to the case of **PM** and **PM-v**, with the visually-augmented HIN, **xPM-v** performed better than **xPM** and even outperformed **GA** in terms of MAP@N. This depicts how the proposed explainable approach can effectively utilize visual information and improve Accuracy in terms of MAP@N. As for **Amazon** dataset, the results are in Figure 4.7b. From this figure, both **xPM** and **xPM-v** outperformed the other baselines including **GA** and **GA-v** in terms of MAP@N. In terms of Recall@N, they performed similarly to **GA** and **GA-v** while outperforming the others. This suggests that, on **Amazon** dataset, both **xPM** and **xPM-v** can produce accurate recommendations. Comparing the proposed models, **xPM** and **xPM-v** performed similarly to each other in terms of MAP@N and Recall@N on this dataset.

Next, the Explainability performance is discussed. Table 4.5 shows the EF@N results for the proposed models and baselines on both **MovieLens** and **Amazon** datasets. Highlighted in bold are the highest values within each column. On **MovieLens** dataset, for every N, all models exhibited comparable performance, with slight differences for larger values of N,



(a) MovieLens



(b) Amazon

Figure 4.7: Accuracy (MAP@N and Recall@N) comparison between the proposed approaches (with the explainability component), **xPM** and **xPM-v**, and other baselines on (a) **MovieLens** and (b) **Amazon** datasets.

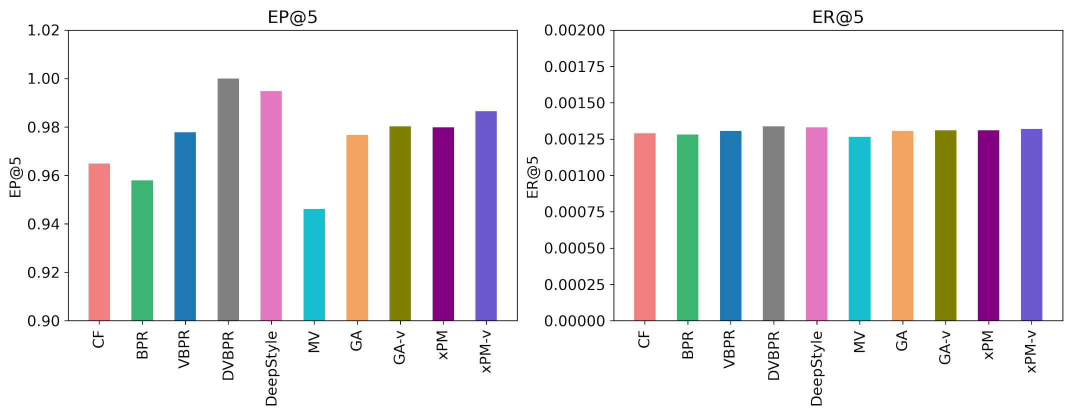
i.e., $N = 10, 50$ and 100. **DVBPR** and **DeepStyle** outperformed other models for $N = 10, 50$ and 100, with **DVBPR** achieving the highest EF@N values. On **Amazon** Dataset, the proposed models, **xPM** and **XPM-v**, and **BPR** consistently outperformed the other models for $N = 5, 10, 50$ and 100. **DVBPR**, **DeepStyle**, **GA**, and **GA-V** also exhibited competitive performance, with their EF@N values being the same or close to the proposed models and **BPR**.

For a more detailed analysis of the Explainability performance, the Explainability Precision and Explainability Recall results are shown in Figure 4.8. The results on **MovieLens** dataset are presented in Figure 4.8a. From this figure, **xPM** performed similarly to **GA** while it outperformed the other non-visually aware baselines including **CF**, **BPR**, and **MV** in terms of EP@5. This indicates that when visual information was not considered, the proposed

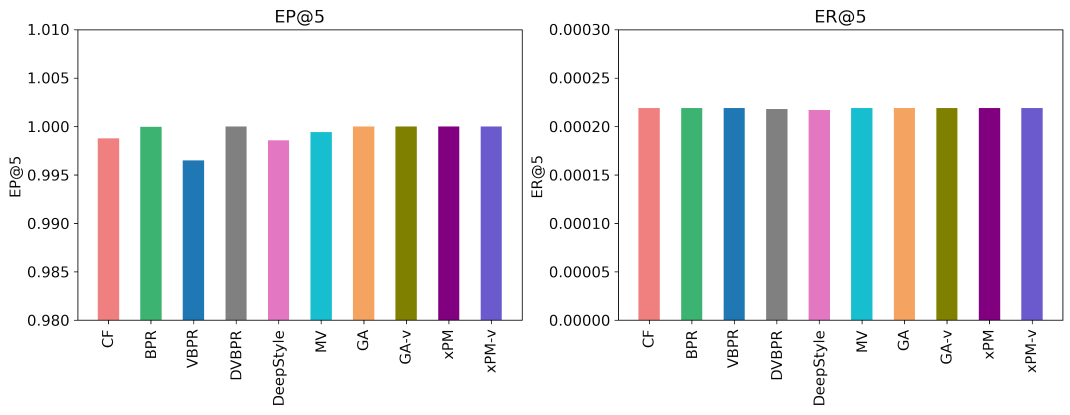
Table 4.5: EF@N results of the baselines and the variations of the proposed approach on **MovieLens** and **Amazon** datasets. The highest values in each column are highlighted in bold.

Model	MovieLens					Amazon				
	F1@1	F1@5	F1@10	F1@50	F1@100	F1@1	F1@5	F1@10	F1@50	F1@100
CF	0.00060	0.00260	0.00519	0.02033	0.03058	8.7996E-05	**4.3790E-04**	8.7562E-04	2.7699E-03	2.7679E-03
BPR	0.00060	0.00260	0.00519	0.02566	0.05084	8.7996E-05	**4.3790E-04**	**8.7761E-04**	**4.3764E-03**	**8.7337E-03**
VBPR	0.00060	0.00260	0.00519	0.02585	0.05123	8.7996E-05	**4.3790E-04**	8.7362E-04	4.3544E-03	8.6839E-03
DVBPR	0.00060	0.00260	**0.00539**	**0.02645**	**0.05201**	8.7996E-05	4.3590E-04	8.6762E-04	4.3406E-03	8.6643E-03
DeepStyle	0.00060	0.00260	**0.00539**	0.02605	0.05162	8.5996E-05	4.3391E-04	8.6962E-04	4.3346E-03	8.6503E-03
MV	0.00060	0.00260	0.00499	0.02487	0.04656	8.7996E-05	**4.3790E-04**	8.7562E-04	3.9374E-03	5.6671E-03
GA	0.00060	0.00260	0.00519	0.02585	0.05064	8.7996E-05	**4.3790E-04**	**8.7761E-04**	**4.3764E-03**	8.6461E-03
GA-V	0.00060	0.00260	0.00519	0.02585	0.05064	8.7996E-05	**4.3790E-04**	**8.7761E-04**	**4.3764E-03**	8.6461E-03
xPM	0.00060	0.00260	0.00519	0.02585	0.05084	8.7996E-05	**4.3790E-04**	**8.7761E-04**	**4.3764E-03**	**8.7337E-03**
xPM-v	0.00060	0.00260	0.00519	0.02585	0.05103	8.7996E-05	**4.3790E-04**	**8.7761E-04**	**4.3764E-03**	**8.7337E-03**

model **xPM** achieved higher Explainability than the other baselines. **xPM** also performed similarly to **VBPR** in terms of EP@5. However, compared to **DVBPR** and **Deepstyle**, **xPM** performed clearly worse than both of these models. This suggests that, compared to the baselines involving visual information, the proposed model without considering visual information is less effective in terms of Explainability. **xPM-v** performed better than most baselines except **DVBPR** and **DeepStyle** in terms of EP@5. This indicates that the proposed approach can produce more explainable items than almost every baseline. Although **DVBPR** and **DeepStyle** performed well regarding Explainability, they clearly performed worse than the proposed approaches in terms of Precision and Recall. This suggests that **xPM-v** maintained a better trade-off between Accuracy and Explainability compared to **DeepStyle**. **xPM-v** outperformed **xPM** in terms of EP@5. This suggests that, when visual information is included, the Explainability performance of the proposed approach can be enhanced. It shows that the proposed approach can achieve high Explainability especially when utilizing visual information on this dataset. In terms of ER@5, all models performed similarly. **DVBPR** and **DeepStyle** slightly outperformed the others. However, the differences are almost indistinguishable. This is because $|\mathcal{E}_u|$ is typically large since there are many items that are explainable for each user. Therefore, ER@N tends to be tremendously small regardless of the number of relevant explainable items in top-N recommendations ($|\mathcal{E}_u \bigcap \mathcal{Y}_u|$). On **Amazon** dataset, the results are shown in Figure 4.8b. From this figure, all models including variations of the proposed approach performed similarly as their $EP@5$ performances are close to each other. One possi-



(a) MovieLens



(b) Amazon

Figure 4.8: Explainability (EP@N and ER@N) comparison between the proposed approaches (with the explainability component), **xPM** and **xPM-v**, and other baselines on (a) **MovieLens** and (b) **Amazon** datasets.

ble reason is that e-commerce purchase behaviors are easier to explain than movie preference rating/tagging behaviors. Most recommendations that were predicted in all of these models are explainable. In terms of ER@5, the results are similar to those on **MovieLens** dataset. Specifically, the ER@5 performances are similar for all the compared approaches. The reason is the same as previously mentioned. Since the number of explainable items for each user is high, the computed ER@N of each user is then typically low, i.e., $|\mathcal{E}_u| >> |\mathcal{E}_u \bigcap \mathcal{Y}_u|$ when computing ER@N.

4.5.3 Scalability Discussion

In this part, the Scalability performance of the proposed approach **xPM-v** compared to the other two baselines that performed well in terms of Accuracy, i.e., **BPR** and **GA**, is discussed.

Table 4.6: The statistics of the synthetic datasets

Synthetic Dataset	Scale	#nodes	#relations
SD1	10^4	4,356	9,986
SD2	10^5	30,902	100,000
SD3	10^6	90,000	1,000,000
SD4	10^7	90,001	10,000,000

The computational time of these approaches on a synthetic dataset was compared. This synthetic dataset consists of 4 sub-datasets namely SD1, SD2, SD3, and SD4. Each sub-dataset contains a different number of relations at a different scale, i.e., 10^4, 10^5, 10^6, and 10^7. The statistics of these sub-datasets are shown in Table 4.6. For all models, the training batch size was set to 16 and they were trained for 10 epochs. The results are in Figure 4.9. From this figure, the computational time of **xPM-v** is slightly higher than **BPR** because **xPM-v** requires additional time for computing the meta-path features and the meta-path based explainability scores and updating the additional parameters in the modified BPR-MF framework. However, compared to **GA**, the computational time of **xPM-v** is much lower, especially for those large-scaled datasets (10^6 and 10^7). These results suggest that the proposed model **xPM-v** achieved close or similar performances with the popular shallow model **BPR**, from the aspect of Scalability. Also, it has significantly higher Scalability than the deep learning model **GA**.

To examine the Scalability of using probabilistic meta-paths and the meta-path based explainability, the computational time of **PM**, **PM-v**, **xPM** and **xPM-v** were compared. The results are shown in Figure 4.9. From this figure, all of these variations have similar computational time with the differences for the sub-dataset SD4, the largest one. However, these differences are not as vast as the difference between **xPM-v** and **GA** discussed previously. Thus, using probabilistic meta-paths for computing the meta-path features and incorporating the meta-path based explainability can be considered scalable in this experiment.

Scalability and Accuracy of using the local connectivity (Eq. (4.5) and (4.6)) and the global connectivity (Eq. (4.9) and (4.12)) were also compared. Given a set of meta-paths $\{UP, UPUP, UPUPUP\}$ based on the user-item interaction node type, the computational time of each method used for computing the meta-path features on the synthetic dataset was examined. These meta-paths were chosen because they are basic meta-paths that can be

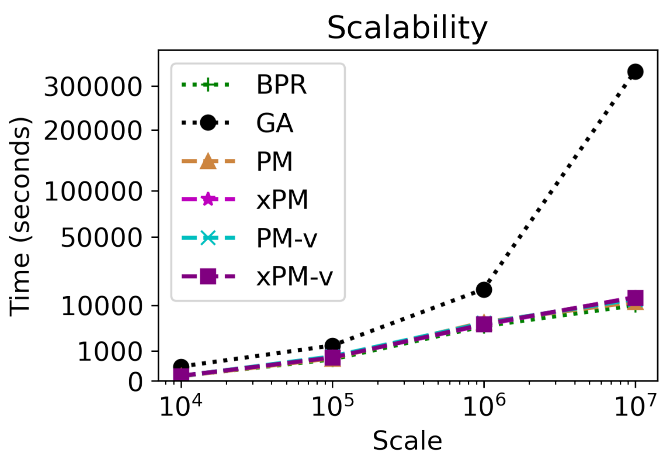


Figure 4.9: Scalability comparison between the proposed model variations, including **PM**, **xPM-v**, **PM-v**, and **xPM-v**), against well-performing baselines in terms of Accuracy, i.e., **BPR** and **GA**). It compares the computational time of each model in seconds.

applied to different datasets. Any dataset generally consists of user and item node types. Some semantic node types may not be present across various datasets. For example, the actor node type in a movie-domain dataset is not available in a clothing-domain dataset. The results are shown in Figure 4.10. This figure shows that the proposed global connectivity method spent less computational time compared to the local connectivity method. This demonstrates the scalability of computing meta-path features using the global connectivity. Furthermore, the Accuracy performance of the proposed approach using the global connectivity and the local connectivity was also examined. This is to validate whether using the global connectivity in the proposed approach affects the Accuracy or not. For this experiment, **MovieLens** dataset was used with three meta-paths $\{UP, UPUP, UPUPUP\}$ for meta-path feature extraction. The Accuracy results of **xPM-v** based on the local connectivity and the global connectivity are shown in Figure 4.11. From this figure, both MAP@N and Recall@N of **xPM-v** using the local and global connectivity are similar. This suggests that using the proposed global connectivity in the proposed approach is as accurate as using the local connectivity, but more scalable.

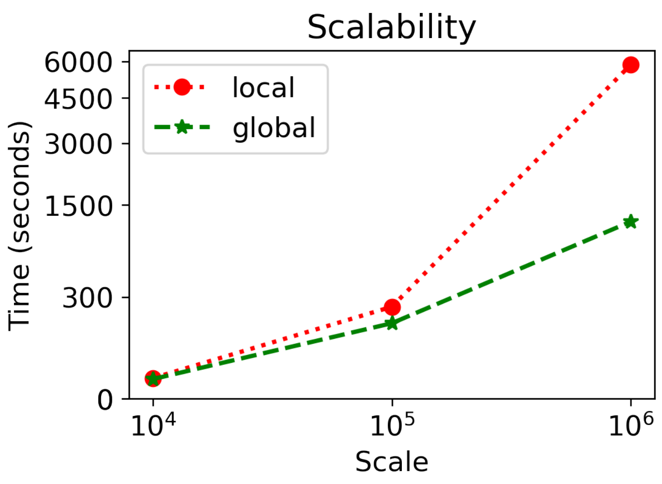
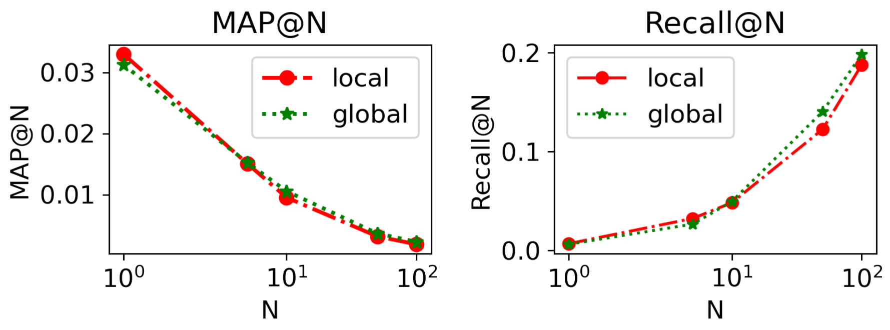
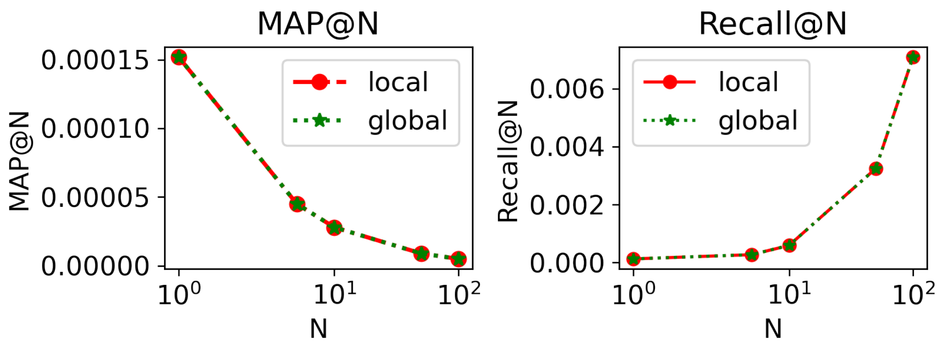


Figure 4.10: Scalability comparison between **xPM-v** using global and **xPM-v** using local connectivity. It compared the computational time of each model in seconds.



(a) MovieLens



(b) Amazon

Figure 4.11: Comparison of Accuracy performance (MAP@N and Recall@N) of **xPM-v** using the local and **xPM-v** using the global connectivity. It shows that using the global connectivity to improve the Scalability performance in **xPM-v** does not drastically affect the Accuracy performance.

4.6 Summary

This chapter focused on the second research question regarding the explainability of visually-aware recommendations using visually-augmented HINs. To increase the explainability of visually-aware recommendations, multi-hop relations are valuable information that was typically ignored in previous studies. However, leveraging multi-hop relations in HINs can pose scalability challenges. To address these challenges and fill the gap of using multi-hop relations for generating visually-aware recommendations, this thesis proposed a scalable and explainable visually-aware recommender system framework called SEV-RS. The framework addressed the limitations of existing approaches by incorporating visual factors, considering scalability issues, and providing explainability in visually-aware recommendations. It consisted of three components: a visually-augmented HIN, a scalable meta-path feature extraction method, and an explainable recommendation generation method that utilized meta-paths.

SEV-RS was evaluated in the Top-N recommendation task using three evaluation metrics: Accuracy, Explainability, and Scalability. The experimental results demonstrated that SEV-RS outperformed baselines in terms of Accuracy, with higher Precision and Recall values. It also effectively leveraged visual information in visually-augmented HINs compared to the Graph Attention Network model. In terms of Explainability, SEV-RS generated more explainable recommendations as indicated by higher Explainability Precision and Explainability Recall values. In terms of Scalability, SEV-RS achieved comparable performance to the shallow BPR-MF model while requiring significantly less computational time compared to the Graph Attention Network model.

Overall, this chapter contributes to the development of a framework for explainable visually-aware recommendations. This framework, along with other meta-path based recommendation approaches, generates explainable recommendations by utilizing meta-paths that convey specific meanings in multi-hop relations. However, the challenge of providing clear and understandable explanations arises when recommendations rely on long and complicated meta-paths. Hence, the third research question of this thesis focuses on enhancing the explainability of meta-path based recommender systems. The following chapter will address this research question and propose a solution to tackle this challenge. It will provide a solution on how to enhance the explainability of HIN-based recommender systems, with a specific focus on meta-path based recommender systems.

Chapter 5

Explainable Recommendations Using Meta-Path Translation

This chapter focuses on the third research question which is about further explainability improvement of explainable visually-aware recommender systems based on HINs. In Chapter 4, a scalable and explainable visually-aware recommender system framework was proposed. This framework uses meta-paths to allow for explainability in visually-aware recommendations. This framework and other meta-path based frameworks can generate recommendations based on specific meta-paths. These meta-paths can be used as explanations for the generated recommendations [69, 72, 96]. As a result, the "explainability" of meta-paths used for learning recommendations in the framework directly affects the explainability of the recommendations. This chapter addresses this issue and aims to provide a method to improve the explainability of not only the proposed framework in the previous chapter but also other HIN-based recommender systems in general.

5.1 Motivation

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. In some cases, long meta-paths are more informative than shorter meta-paths [205, 211, 212, 213]. Despite the effectiveness of long and complicated mete-paths, the explainability of these meta-paths is intuitively low. Ex-

This chapter has been published in [210] available at `https://doi.org/10.1145/3625828`

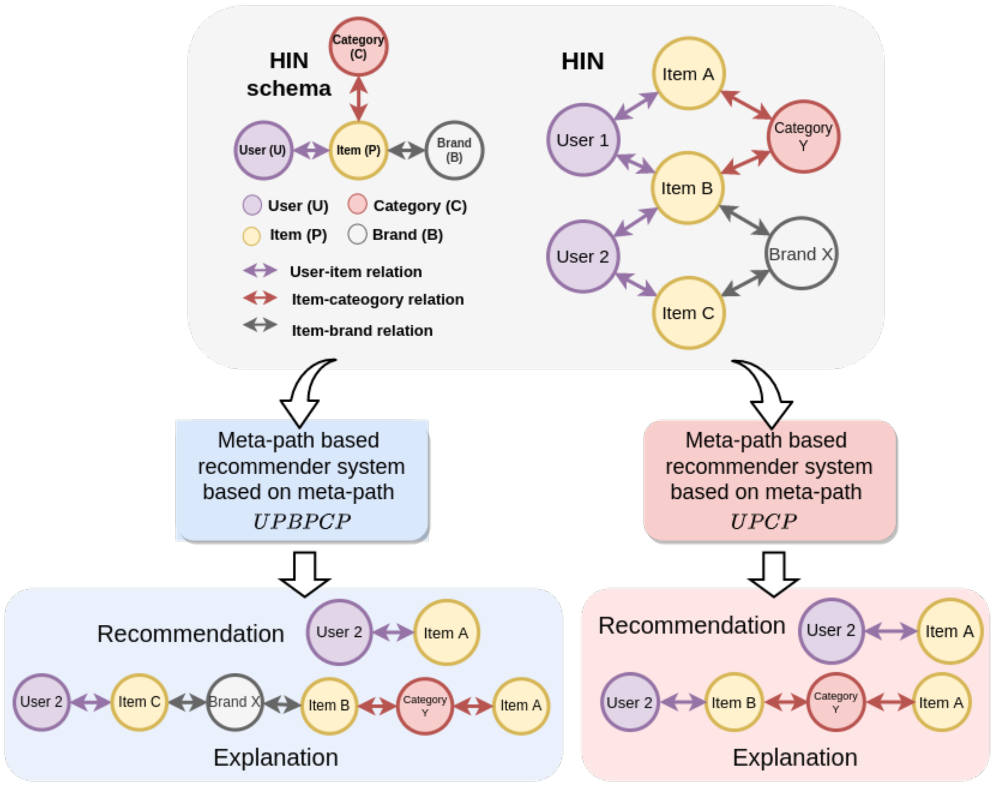


Figure 5.1: Example of explaining recommendations obtained from meta-path based recommender systems using meta-path $UPBPCP$ and meta-path $UPCP$ where U, P, B, and C denote the user, item, brand, and category node type, respectively.

plaining recommendations with these meta-paths could be more difficult to understand than shorter and less complex ones. For example, given a HIN in Figure 5.1, 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 $UPBPCP$. 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, a method to find more comprehensible explanations for meta-path based recommendations is proposed. First, how to quantify **meta-path explainability** based on three aspects, i.e., readability, credibility, and diversity is introduced. Then, given any meta-path, **comparable explainable meta-paths** are defined as meta-paths that perform recommendation similarly but have higher explainability compared to the given 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, a method for generating comparable explainable meta-paths is proposed. Considering meta-paths as languages, it can be assumed that they can be constructed based on certain grammar rules. The **meta-path grammar** is therefore defined based on the concept of quasi-synchronous context-free grammar [214]. With this grammar, a **meta-path translation model** is proposed. This model is a probabilistic model that maps a meta-path to its comparable explainable ones. Since the task of meta-path translation 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 [215]. 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 the meta-path translation task which is a one-to-many task compared to other Seq2Seq models. To the best of current knowledge, this is the first work that considers meta-paths from a linguistic point of view. In summary, the contributions can be outlined as follows:

- The meta-path translation task for recommender systems is introduced, aiming to identify the comparable explainable meta-paths of a long and complicated one.
- The quantification of meta-path explainability is proposed, allowing for the identification of comparable explainable meta-paths for any given meta-path.
- Two meta-path translation datasets were generated based on two real-world recommendation datasets. They are publicly available and represent the first of their kind.

- A novel view of meta-paths and their formation from a grammatical perspective is introduced. A meta-path grammar is defined to enable the systematic construction of explainable meta-paths.
- A meta-path translation model is proposed, leveraging the meta-path grammar defined in this work. This model considers both non-hierarchical and hierarchical structures of a meta-path simultaneously to effectively learn the translation.
- Extensive experiments on real-world datasets were conducted to generate more comprehensible explanations for meta-path based recommendations.

The organization of the remainder of this chapter is as follows: In Section 5.2, the proposed definitions of meta-path explainability and explainable meta-path are introduced. Then, the proposed method to identify comparable explainable meta-paths given a long and complicated one is explained. In Section 5.3, the formulation of the proposed meta-path translation task is discussed. Following that, the meta-path grammar and the proposed meta-path translation model based on this grammar are examined. Subsequently, in Section 5.4, how to construct a meta-path translation dataset for training and evaluating the proposed model is described. Section 5.5 discusses the experiments conducted to evaluate the proposed meta-path translation model. The details of parameterization and evaluation metrics can be found in this section. In Section 5.6, the results and discussions of the experiments are provided. The performance comparisons between the proposed meta-path translation model and state-of-the-art Seq2Seq models are discussed, along with the analysis of hyperparameters, errors, and computational complexity. Lastly, Section 5.7 provides a summary of contributions and conclusions for this chapter.

5.2 Explainable Meta-Path

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. In spite of that, these meta-paths are still useful for capturing diversity in users' preferences [173]. In this section, the definition of an explainable meta-path is introduced. Then, how to identify relatively more explainable meta-paths given a long and complicated one is explained.

Three metrics in [87] are customized to quantify the explainability of a meta-path. These metrics are commonly used to measure sequence (or path) readability, credibility, and diversity. Readability indicates how easily interpretable or understandable a sequence is. Higher readability indicates a more straightforward and intuitive sequence. Credibility is a metric that evaluates the reliability or trustworthiness of a sequence. It measures the degree to which the connections or relationships between the components in a sequence are supported by evidence or data. A higher credibility score corresponds with stronger empirical or contextual support for the sequence. Diversity quantifies the variety or distinctiveness of the components in a sequence. A higher diversity score suggests that the sequence incorporates a broader range of entities, potentially providing more comprehensive information. These metrics are also applicable in meta-paths and can be used to measure similar qualities. In this work, the definitions of meta-path readability, meta-path credibility, and meta-path diversity are introduced. Let $m = N_1 N_2 \cdots N_{l+1}$ be a meta-path. l is the length of m. *Meta-path readability* $\mathrm{R}(m)$ is computed by

$$\mathrm{R}(m) = \frac{1}{\sqrt{l \cdot |\mathbb{N}_m|}} \tag{5.1}$$

where $\mathbb{N}_m = \{N_1, N_2, ..., N_{l+1}\}$ is a set of node types in meta-path m. 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

$$\mathrm{C}(m) = \Pi_{i=1}^{l} \mathbb{W}(R_{N_i,N_{i+1}}) \tag{5.2}$$

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

$$\mathrm{D}(m) = \log_{l+1} |\mathbb{R}_m| \tag{5.3}$$

Some studies have shown that relation sequences with diversity are more comprehensive and persuasive for humans [87, 216]. Thus, the higher the diversity of m is, the more explainable.

Table 5.1 shows an example of various meta-paths consisting only of U and P node types with different lengths and their corresponding readability and diversity values. Intuitively, the

Table 5.1: Example of various meta-paths consisting only of U and P node types with different lengths and their corresponding readability and diversity values

Meta-path	Readability	Diversity
$UPUP$	0.35	0.5
$UPUPUP$	0.29	0.39
$UPUPUPUP$	0.25	0.33

Table 5.2: Example of various meta-paths, each with the same length but differing in the number of distinctive nodes and relation types, and their corresponding readability and diversity values

Meta-path	Readability	Diversity
$UPUPUPUP$	0.25	0.33
$UPTPGPUP$	0.18	0.86
$UPTPGPVP$	0.16	0.94

longer the meta-path, the less explainable it becomes, as it is more complicated to understand. From this table, it is evident that as the meta-path length increases, readability and diversity decrease, consequently leading to lower meta-path explainability. Meanwhile, Table 5.2 shows an example of various meta-paths, each with the same length but differing in the number of distinctive nodes and relation types. This table reveals that when a meta-path incorporates a greater variety of node and relation types, readability tends to decrease while diversity increases. Considering readability, if a meta-path consists of repetitive nodes and relation types, users (or readers) can more easily grasp the underlying meaning. This is because users do not have to understand and distinguish the different meanings of various types of nodes and relations. However, in such cases, a meta-path will typically have low diversity. As aforementioned, a sequence containing diverse information also has the potential to enhance human understanding. Therefore, an explainable meta-path in this thesis should maintain the tradeoff between readability and diversity.

In terms of credibility, the explainability of a meta-path depends on the weights assigned to relation types. These weights can be assigned by experts or based on prior knowledge, indicating how certain relation types are more explainable than others. By assigning high weights to such relation types, meta-paths containing them will achieve higher explainability compared to those without.

Definition 5.1. (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.*

This work proposes 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 5.2. (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*
- $\mathbb{N}_{m^*} \subseteq \mathbb{N}_m$
- $l' \leq \kappa < l$
- $|\mathrm{A}(m) - \mathrm{A}(m^*)| < \delta$

where $\mathbb{N}_{m^*}$ is the set of node types of m^*, l' is the length of m^*, κ is the maximum length for selecting explainable 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, it is possible to 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, a method to find comparable explainable meta-paths for recommendations based on long/complicated meta-paths is proposed.

5.3 Meta-Path Translation

5.3.1 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 $f : \mathcal{M} \to \mathrm{P}(\mathcal{M}^*)$ 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.

5.3.2 Meta-Path Grammar

To find f, properties of meta-paths in $\mathcal{M}$ and $\mathcal{M}^*$ are first assumed. Inspired by the concept of context-free grammar in NLP, it is assumed that meta-paths in $\mathcal{M}$ and $\mathcal{M}^*$ follow certain grammar. Such a grammar represents certain rules of how meta-paths in both $\mathcal{M}$ and $\mathcal{M}^*$ are constructed. Based on Definition 5.2, 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, the concept of *quasi-synchronous context-free grammar* [214] is adopted to define the grammar of meta-paths in $\mathcal{M}$ and $\mathcal{M}^*$.

Definition 5.3. (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*

$$\mathbb{G}[t] = (\mathbb{S}, \mathbb{N}, \mathbb{P}, \mathbb{E}, \mathbb{R}[t], \theta) \tag{5.4}$$

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:

$$\mathbb{S} \to \mathsf{A}[\alpha_i],\ \mathsf{A} \in \mathbb{N}, \alpha_i \subseteq t \tag{5.5}$$

$$\mathsf{A}[\alpha_i] \to \mathsf{B}[\alpha_j]\mathsf{C}[\alpha_k],\ \mathsf{A} \in \mathbb{N}, \mathsf{B}, \mathsf{C} \in \mathbb{N} \cup \mathbb{P}, \alpha_i, \alpha_j, \alpha_k \subseteq t \tag{5.6}$$

$$\mathsf{D}[\alpha_i] \to w,\ \mathsf{D} \in \mathbb{P}, w \in \mathbb{E}, \alpha_i \subseteq t \tag{5.7}$$

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

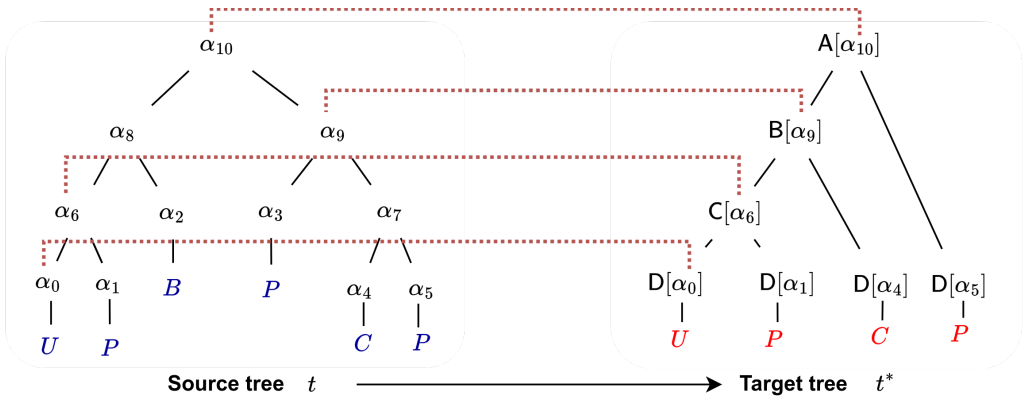


Figure 5.2: Example of meta-path grammar and how the target tree nodes are aligned with certain nodes in the source tree in order to translate the source meta-path $UPBPCP$ to the target meta-path $UPCP$.

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. 5.7 and can expands to a terminal* $w \in \mathbb{E}$ *following the grammar rule* B$[\alpha_i] \rightarrow w$ *(or* C$[\alpha_i] \rightarrow w$*).*

Example 5.1. (Meta-Path Grammar) Figure 5.2 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 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.

5.3.3 Meta-Path Translation Model

In this section, the proposed model to translate or map a given meta-path to its comparable explainable meta-paths is discussed. To build a meta-path translation model, a Seq2Seq model with latent neural grammar from [215] is adopted. 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 the 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 5.3 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, there is no parser developed specifically for meta-paths. Therefore, the same approach in [215] is adopted to jointly train the parser with the encoder and the decoder. Following [215], 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 used to obtain dependency infor-

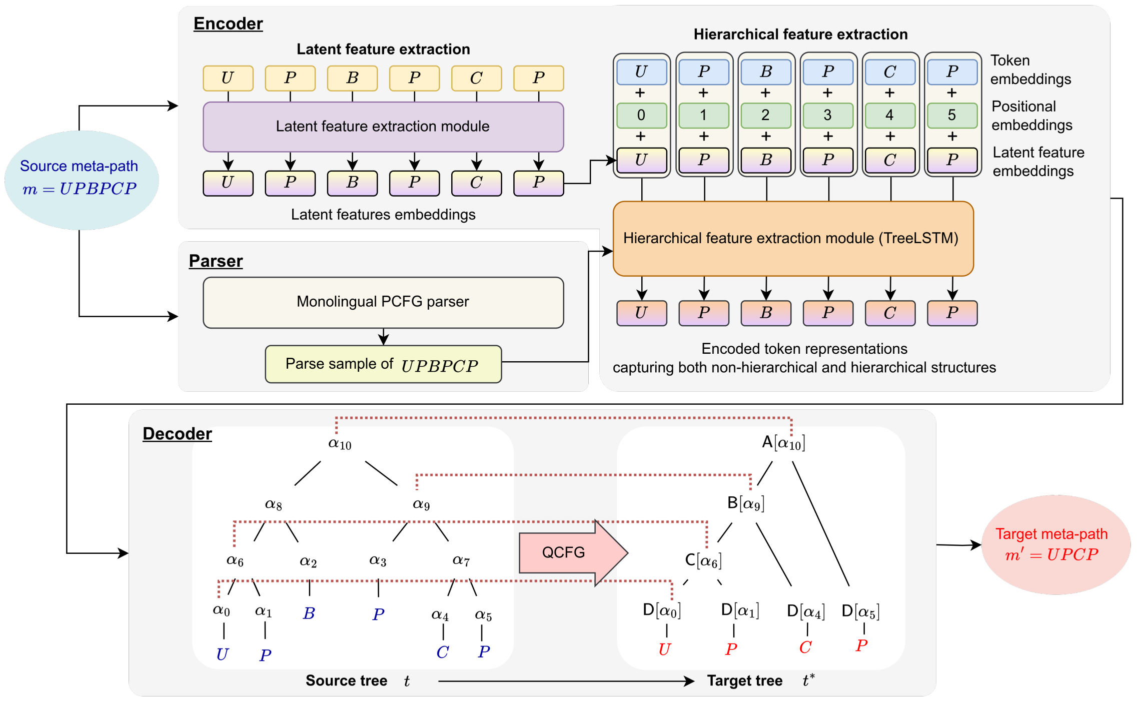

Encoder
Latent feature extraction
Latent feature extraction module
Latent features embeddings
Hierarchical feature extraction
Token embeddings
Positional embeddings
Latent feature embeddings
Hierarchical feature extraction module (TreeLSTM)
Encoded token representations capturing both non-hierarchical 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 $m' = UPCP$

Figure 5.3: The proposed meta-path translation model which comprises (1) a parser parsing a source meta-path and generating the source tree, (2) an encoder consisting of latent feature extraction module for extracting dependency information of node types in a source meta-path regardless of its hierarchical structure and hierarchical feature extraction module for encoding the hierarchical structure of a source meta-path given its source tree from the parser, and (3) a decoder transducing the source tree to the target tree.

mation 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. 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 [215], TreeLSTM [217] 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, the same node types repeatedly appear in a meta-path, for instance, $UPBPCP$ has three P node types in three different positions. This could lead to confusion in learning the dependency information. Thus, the *positional embeddings* are added 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:

$$\mathbf{z}_{\alpha_i} = \sigma\left(\mathbf{W}^{(\mathbf{z})}(\mathbf{x}_{\alpha_i} + \mathbf{x}^*_{\alpha_i} + \mathbf{x'}_{\alpha_i}) + \sum_{n=1}^{N} \mathbf{U}_n^{(\mathbf{z})}\mathbf{h}_{\alpha_i n} + b^{(\mathbf{z})}\right) \tag{5.8}$$

$$\mathbf{f}_{\alpha_i k} = \sigma\left(\mathbf{W}^{(\mathbf{f})}(\mathbf{x}_{\alpha_i} + \mathbf{x}^*_{\alpha_i} + \mathbf{x'}_{\alpha_i}) + \sum_{n=1}^{N} \mathbf{U}_{kn}^{(\mathbf{f})}\mathbf{h}_{\alpha_i n} + b^{(\mathbf{f})}\right) \tag{5.9}$$

$$\mathbf{o}_{\alpha_i} = \sigma\left(\mathbf{W}^{(\mathbf{o})}(\mathbf{x}_{\alpha_i} + \mathbf{x}^*_{\alpha_i} + \mathbf{x'}_{\alpha_i}) + \sum_{n=1}^{N} \mathbf{U}_n^{(\mathbf{o})}\mathbf{h}_{\alpha_i n} + b^{(\mathbf{o})}\right) \tag{5.10}$$

$$\mathbf{v}_{\alpha_i} = \tanh\left(\mathbf{W}^{(\mathbf{v})}(\mathbf{x}_{\alpha_i} + \mathbf{x}^*_{\alpha_i} + \mathbf{x'}_{\alpha_i}) + \sum_{n=1}^{N} \mathbf{U}_n^{(\mathbf{v})}\mathbf{h}_{\alpha_i n} + b^{(\mathbf{v})}\right) \tag{5.11}$$

$$\mathbf{c}_{\alpha_i} = \mathbf{z}_{\alpha_i} \odot \mathbf{v}_{\alpha_i} + \sum_{k=1}^{N} \mathbf{f}_{\alpha_i k} \odot \mathbf{c}_{\alpha_i k} \tag{5.12}$$

$$\mathbf{h}_{\alpha_i} = \mathbf{o}_{\alpha_i} \odot \tanh(\mathbf{c}_{\alpha_i}) \tag{5.13}$$

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, σ 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:

$$p_\theta(\$ \to \mathsf{A}[\alpha_i]) = \varsigma(\mathbf{u}_\$^T \mathbf{e}_{\mathsf{A}[\alpha_i]}) \tag{5.14}$$

$$p_\theta(\mathsf{A}[\alpha_i] \to \mathsf{B}[\alpha_j]\mathsf{C}[\alpha_k]) = \varsigma(f_1(\mathbf{e}_{\mathsf{A}[\alpha_i]})^T f_2(\mathbf{e}_{\mathsf{B}[\alpha_j]}) + f_3(\mathbf{e}_{\mathsf{C}[\alpha_k]})) \tag{5.15}$$

$$p_\theta(\mathsf{D}[\alpha_i] \to w) = \varsigma(f_4(\mathbf{e}_{\mathsf{D}[\alpha_i]})^T \mathbf{u}_w + b_w) \tag{5.16}$$

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

$$\mathbf{e}_{\mathsf{A}[\alpha_i]} = \mathbf{u}_{\mathsf{A}} + \mathbf{h}_{\alpha_i} \tag{5.17}$$

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

$$\log p(m^*|m) = \log \Big(\sum_{t\in\mathcal{T}(m)} \sum_{t^*\in\mathcal{T}(m^*)} p_\theta(t^*|t) p_\phi(t|m) \Big) \tag{5.18}$$

where $\mathcal{T}(m)$ and $\mathcal{T}(m^*)$ denote the sets of trees whose yields are m and m^* respectively. Since

$$\sum_{t^*\in\mathcal{T}(m^*)} p_\theta(t^*|t) = p_\theta(m^*|t), \tag{5.19}$$

Eq. (5.18) can be rewritten as

$$\begin{aligned} \log p(m^*|m) &= \log \Big(\sum_{t\in\mathcal{T}(m)} p_\theta(m^*|t) p_\phi(t|m) \Big) \\ \log p(m^*|m) &= \log \Big(\mathbb{E}_{t\sim p_\phi(t|m)} [p_\theta(m^*|t)] \Big) \end{aligned} \tag{5.20}$$

where

$$p_\theta(m^*|t) = \prod_{r\in\mathbb{R}[t]} p_\theta(r) \tag{5.21}$$

where $\mathbb{R}[t]$ denotes a set of rules in the source tree t and $p_\theta(r)$ is the probability of rule r. By Jensen's inequality,

$$\log p(m^*|m) \geq \mathbb{E}_{t\sim p_\phi(t|m)} [\log p_\theta(m^*|t)] \tag{5.22}$$

as the lower bound of the log-likelihood. The greatest lower bound of $\log p(m^*|m)$ can be estimated by maximizing this lower bound. Thus, the loss function is defined as an expected negative log-likelihood, Finally, the loss function is defined as an expected negative log-likelihood,

$$L(\theta, \phi) = -\,\mathbb{E}_{t\sim p_\phi(t|m)} [\log p_\theta(m^*|t)] \tag{5.23}$$

In this work, unlike the original model in [215], the regularization terms are also added in the loss function to ensure the explainability of meta-paths as follows:

$$L(\theta, \phi) = -\mathbb{E}_{t \sim p_\phi(t|m)}\big[\text{log } p_\theta(m^*|t)\big] - \lambda\big(\mathrm{R}(m^*) + \mathrm{C}(m^*) + \mathrm{D}(m^*)\big)$$

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 5.3 illustrates the whole proposed framework. This figure shows an example 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 A$[\alpha_{10}]$ in t^* is transduced by α_{10} in t. The grammar rule A$[\alpha_{10}] \rightarrow$ B$[\alpha_9]$D$[\alpha_5]$ is learned by the decoder. This means that, in the target tree t^*, A$[\alpha_{10}]$ is parsed/divided into B$[\alpha_9]$ which is transduced by α_9 and D$[\alpha_5]$ which is transduced by α_5.

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 back-propagated through the usual inside algorithm [218]. For the gradient with respect to ϕ, the score function estimator with a self-critical baseline [219] is applied as follows:

$$\nabla_\phi L(\theta, \phi) \approx \ -(g(\tilde{t}) - g(\hat{t}))\nabla_\phi \text{log } p_\phi(\tilde{t}|m) \tag{5.24}$$

where $g(t) = \text{log } p_\theta(m^*|t)$, $\tilde{t}$ 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_t 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.

5.4 Meta-Path Translation Dataset Generation

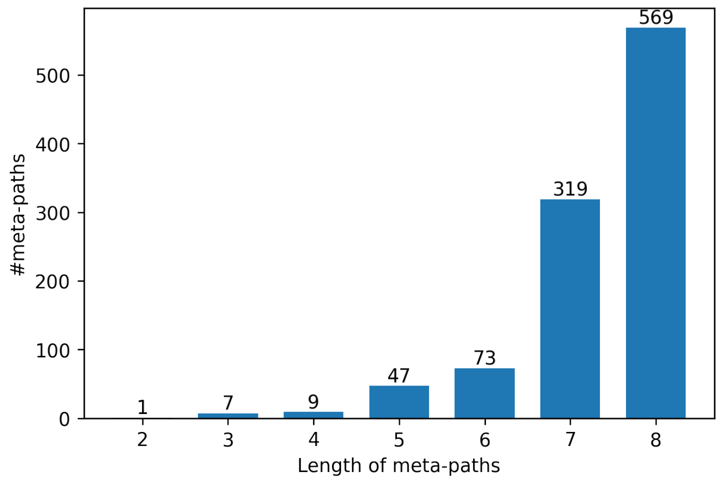
To train the proposed meta-path translation model, a meta-path translation dataset is needed. There are three requirements for generating such a dataset, i.e., a recommendation dataset, a meta-path based recommendation model, and a set of meta-paths $\mathcal{M}$ for being applied to the selected recommendation model on the selected dataset. Given these inputs, the process of generation is summarized as follows:

Step 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.

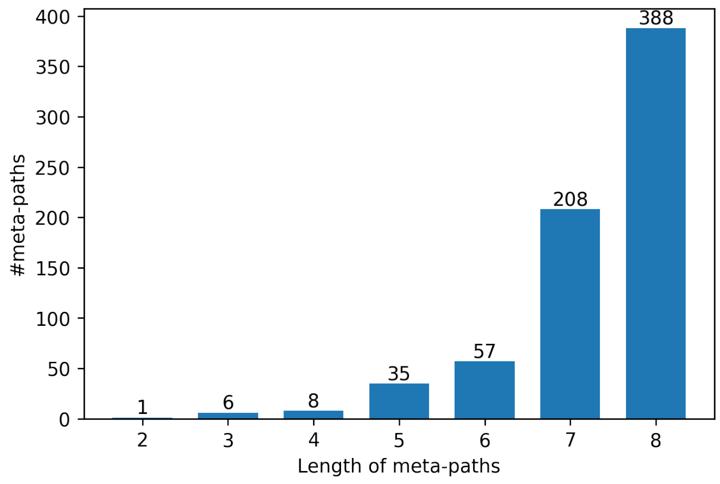
Step 2: With all sets of recommendations based on each and every $m \in \mathcal{M}$, the set of all comparable explainable meta-paths of each m, $f(m)$, is identified. This is done by iteratively considering all possible pairs of meta-paths and validating them based on Definition 5.2.

Step 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 the experiments, the recommendation model in [173] 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. This approach was selected since it is easy to implement and requires less computational resources than most approaches. Two real-world recommendation datasets were used. These datasets are the same datasets as in Chapter 4 (See Section 4.4.1). 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 4.1 (in Section 4.4.1). For **MovieLens**, there are 7 node types and 14 relation types (inverse relation types included). For **Amazon** dataset, there are 6 node types and 12 relation types (inverse relation types included). For simplicity, in this work, all relations have the same weight which is 1, i.e., $w(x, y) = 1$ 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 4.1 (in Section 4.4.1) were selected. Only meta-paths that start with the user node type U and end with the item node type P were considered. 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, this meta-path was discarded. The range of meta-path length then starts at 2. To determine the maximum length, the fact that increasing the maximum length would result in a significantly larger number of meta-paths was considered. For each



(a) MovieLens



(b) Amazon

Figure 5.4: The number of meta-paths of each length ranging from 2 to 8 used in the experiments on (a) **MovieLens** and (b) **Amazon** datasets

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 the experiments while considering the exponential growth of possible meta-paths with increasing length, the maximum length of the selected meta-paths is 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 **Amazon** dataset consists of 703 meta-paths. For both datasets, the number of meta-paths of each length can be found in Figure 5.4.

For Step 2, to identify explainable meta-paths, the maximum length of comparable explainable meta-paths (κ), the readability threshold (δ_R), the credibility threshold (δ_C) and the diversity threshold (δ_D) have to be determined. The parameter κ was varied 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 identical for every meta-path. This means that different values of δ_C did not affect the number of explainable meta-paths. Therefore, only outcomes when varying κ, δ_R, and δ_D were considered. Figures 5.5 and 5.6 show the number of explainable meta-paths when different κ, δ_R and δ_D were used on **MovieLens** dataset and **Amazon** dataset respectively. To ensure explainability, κ should be as short as possible. However, when $\kappa = 2$, there is only one explainable meta-path available for both datasets. Therefore, $\kappa = 3$ was selected, which is the second shortest length for both datasets. Then, to include as many explainable meta-paths as possible, the highest thresholds that would allow the highest number of explainable meta-paths when $\kappa = 3$ was selected, 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 [65, 73, 173, 175, 208, 209, 220, 221, 222, 223, 224], they are corresponding with the majority of the shortest meta-paths that were commonly used. Intuitively, shorter meta-paths are easier to comprehend. Since the chosen meta-paths are the shortest meta-paths commonly used, this observation suggests that they are practical and explainable. From Definition 5.2, at least one performance evaluation metric is required to identify comparable explainable meta-paths. In the 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 5.2 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.

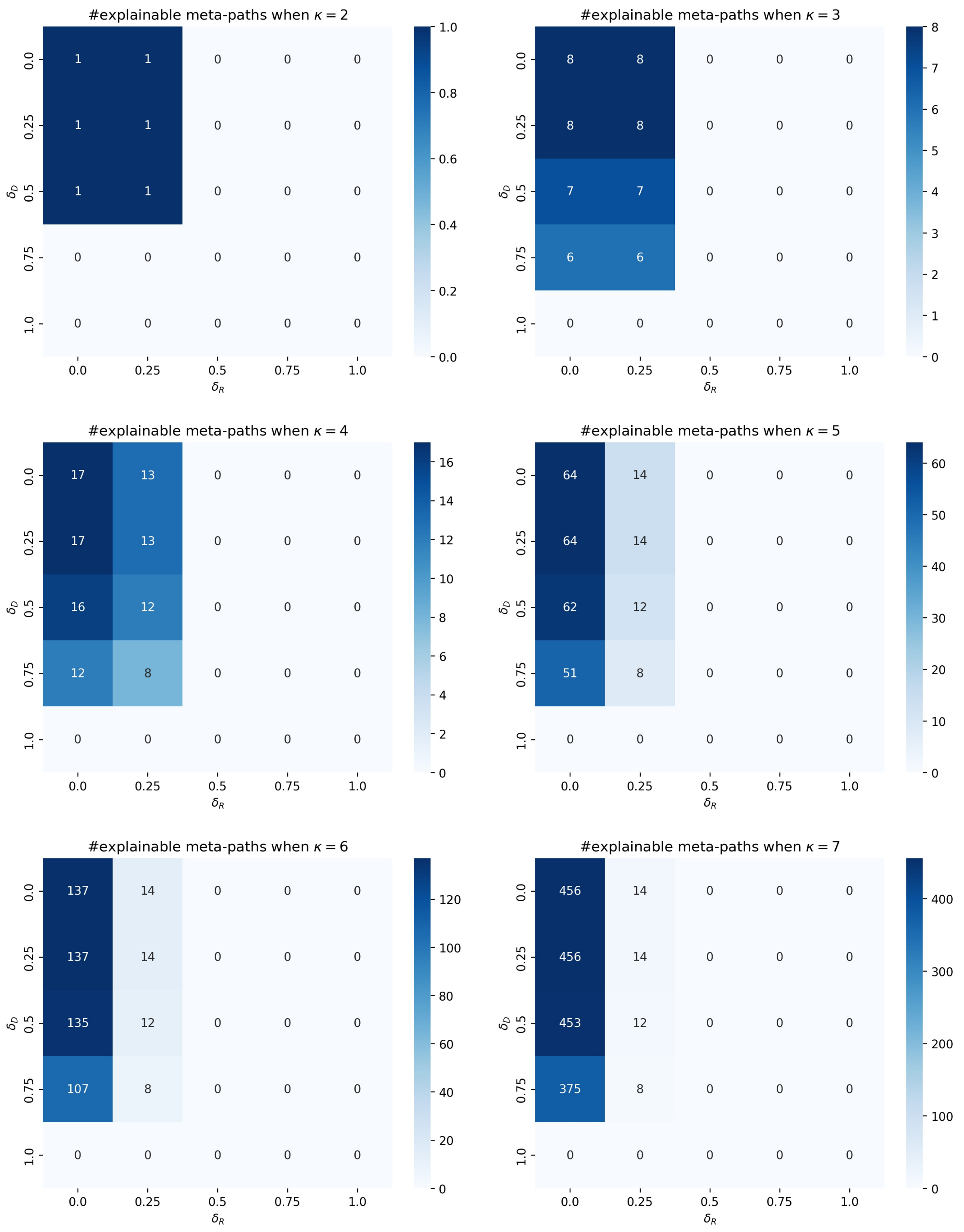


Figure 5.5: The number of explainable meta-paths when different values of the maximum length of comparable explainable meta-paths (κ), the readability threshold (δ_R), and the diversity threshold (δ_D) were used to determine explainable meta-paths on **MovieLens** dataset

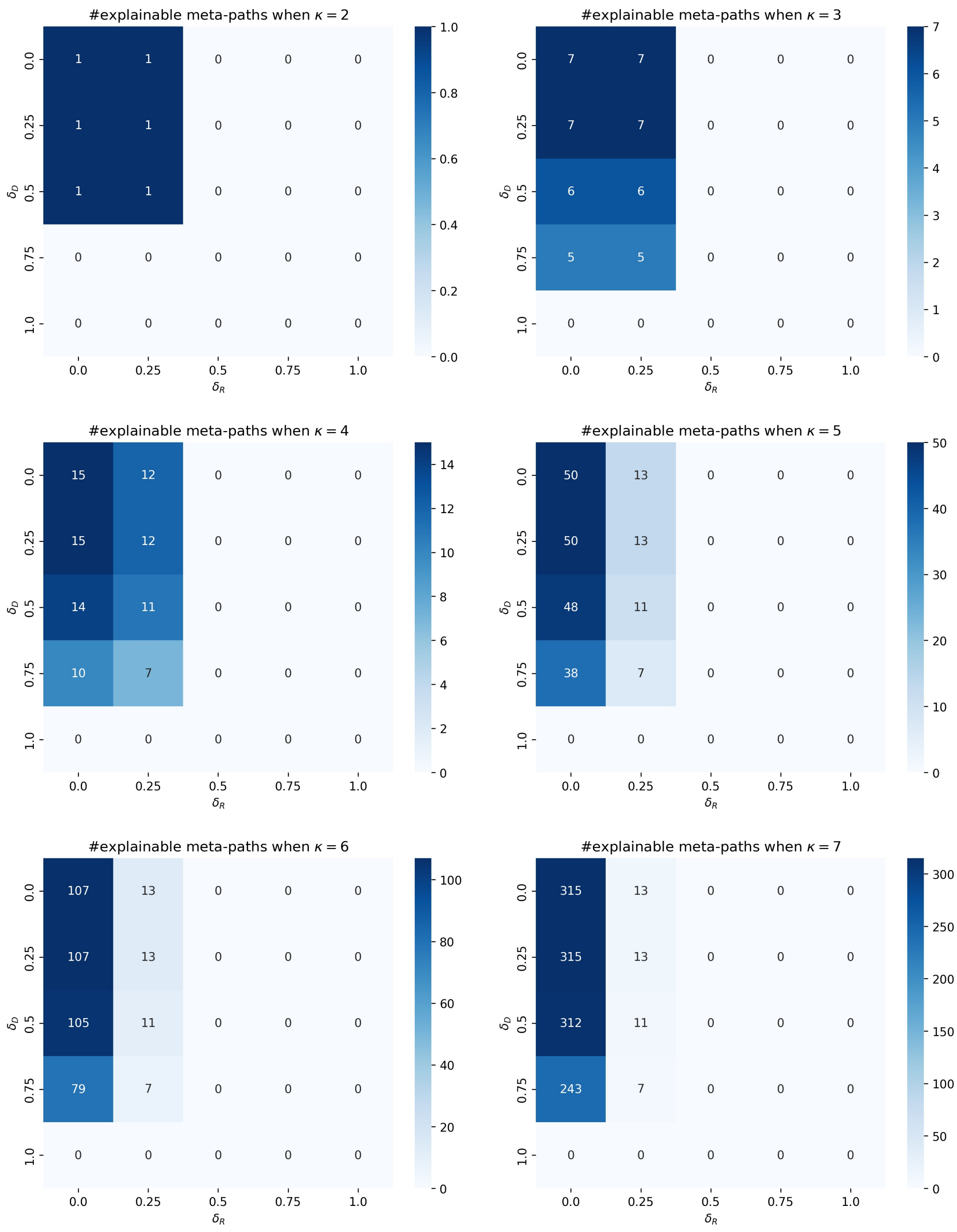
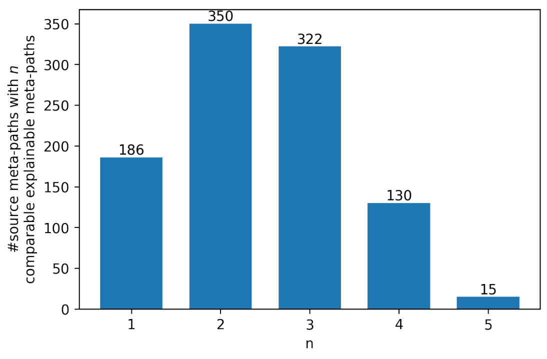
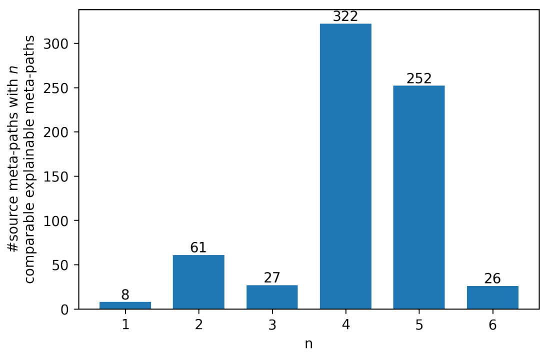


Figure 5.6: The number of explainable meta-paths when different values of the maximum length of comparable explainable meta-paths (κ), the readability threshold (δ_R), and the diversity threshold (δ_D) were used to determine explainable meta-paths on **Amazon** dataset



(a) MovieLens



(b) Amazon

Figure 5.7: The number of source meta-paths with n comparable explainable meta-paths in the meta-path translation datasets based on (a) **MovieLens** and (b) **Amazon** datasets

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 5.7. On average, one source meta-path is corresponding to 2.44 target meta-paths in **MovieLens** dataset and 4.25 target meta-paths in **Amazon** dataset. Table 5.3 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, 70% of the source meta-paths were split for training, and 30% of them were reserved for testing. The statistics of the generated datasets are summarized in Table 5.4. 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 the meta-path translation datasets is less than the number of all possible meta-paths used for predicting recommendations. The datasets and the implementation of the meta-path translation model can be found in this link `https://bit.ly/meta-path-translation-model` with an access request required.

Table 5.3: 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, $UVUP$, $UPBP$, $UPVP$

Table 5.4: The statistics of the meta-path translation datasets generated based on **MovieLens** and **Amazon** datasets

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

5.5 Experimental Setup

Experiments were conducted on the generated datasets previously described in Chapter 5.4. The objective of the experiments is to validate the effectiveness of the proposed meta-path translation model compared with the baselines. An ablation study was conducted to examine the effect of the number of sampled target trees K_t and the weight decay w_d on the performance of the proposed meta-path translation model. Moreover, an error analysis was performed to examine incorrect predictions generated by the proposed model.

5.5.1 Parameterization

As for the proposed meta-path translation model, two deep learning architectures, i.e., CNN [225] and Transformer, [226] 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_t was varied among $\{10, 30, 50, 70, 100\}$, and $K_t = 50$ was used for both models on both datasets.

5.5.2 Evaluation Metrics

Two evaluation metrics, Mean Hit Ratio@N (HR@N) and Mean Recall@N (Recall@N), were used for the comparison of accuracy. HR@N is computed as follows:

$$HR@N = \frac{|\mathcal{M}^S_{hit}|}{|\mathcal{M}^S|} \tag{5.25}$$

where $|\mathcal{M}^S_{hit}|$ is the number of source meta-paths for which one of the correct target meta-paths is included in the top-N predictions and $|\mathcal{M}^S|$ is the total number of source meta-paths. The higher HR@N means the better the model can predict one of the target meta-paths for each source meta-path correctly. However, since there can be multiple corresponding target meta-paths for each source meta-path, Recall@N is used to evaluate the performance of predicting a group of target meta-paths. It is computed as follows:

$$Recall@N = \frac{1}{|\mathcal{M}^S|}\sum_{m\in\mathcal{M}^S}\frac{|\mathcal{P}_m \cap \mathcal{P}^N_m|}{|\mathcal{P}_m|}, \tag{5.26}$$

where $\mathcal{P}_m$ denotes the set of corresponding target meta-paths of source meta-path $m \in \mathcal{M}^S$ and $\mathcal{P}^N_m$ denotes the set of top-N predicted target meta-paths of m.

Furthermore, the explainability of the translated meta-paths was also evaluated by Mean Readability@N (RD@N) and Mean Diversity@N (DV@N). These two metrics indicate the average diversity and readability of the top-N translated meta-paths obtained from the model. They can be computed as follows:

$$RD@N = \frac{1}{N\cdot|\mathcal{M}^S|}\sum_{m\in\mathcal{M}^S}\sum_{m^*\in\mathcal{P}^N_m}\mathrm{R}(m^*) \tag{5.27}$$

and

$$DV@N = \frac{1}{N\cdot|\mathcal{M}^S|}\sum_{m\in\mathcal{M}^S}\sum_{m^*\in\mathcal{P}^N_m}\mathrm{D}(m^*) \tag{5.28}$$

where $\mathrm{R}(m^*)$ denotes the readability of meta-path m^* defined in Eq. 5.1 and $\mathrm{D}(m^*)$ denotes and diversity of meta-path m^* defined in Eq. 5.3. $\sum_{m\in\mathcal{M}^S}\sum_{m^*\in\mathcal{P}^N_m}\mathrm{R}(m^*)$ is the sum of readability of all predicted target meta-paths of every source meta-path. Similarly, $\sum_{m\in\mathcal{M}^S}\sum_{m^*\in\mathcal{P}^N_m}\mathrm{D}(m^*)$ is the sum of the diversity of all predicted target meta-paths of every source meta-path. They are both divided by the total number of predicted target meta-paths which is $N\cdot|\mathcal{M}^S|$ (N predicted target meta-paths for each source meta-path) to find their mean values. Regarding credibility, since all relation weights are identical, the credibility of the translated meta-path was not considered in this work. The use of varied relation weights and the result regarding credibility be investigated further in future work.

5.6 Results and Discussions

Most state-of-the-art machine translation models heavily relied on pre-training on large corpus [227, 228]. Since pre-training is not applicable in this case, the proposed meta-path translation model was compared with some Seq2Seq baselines that do not require pre-training as follows:

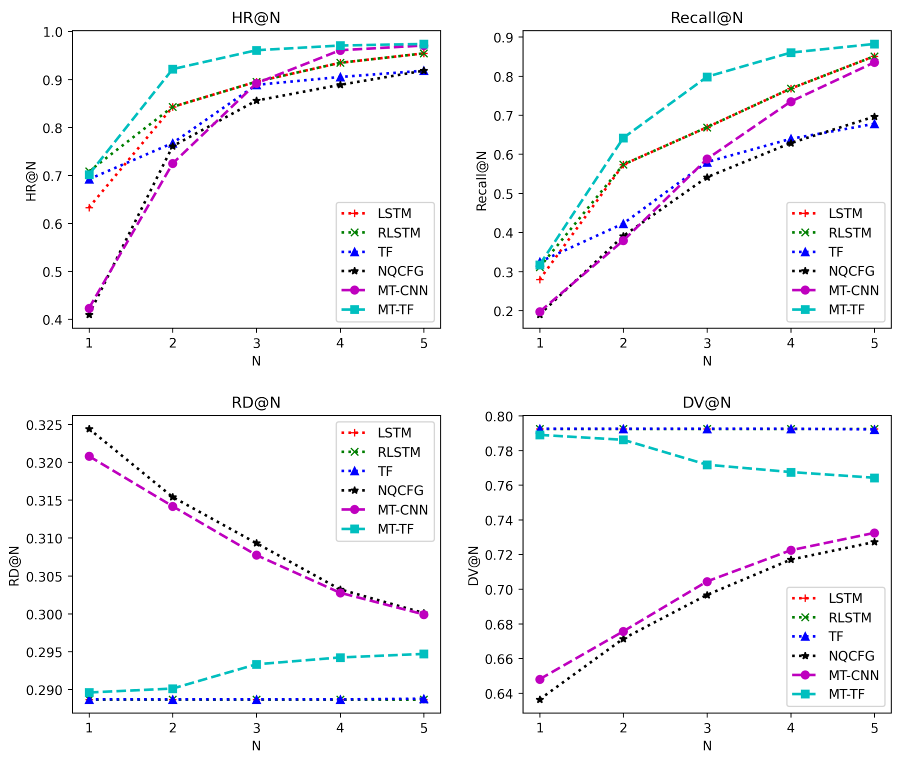
- **Long Short-Term Memory (LSTM)** [229]: a simple Seq2Seq model based on an LSTM network. Both encoder and decoder embedding sizes were set to 16. The model was trained for 50 epochs. The other parameters were set as in the original model.
- **Reverse Long Short-Term Memory (RLSTM)** [230]: a Seq2Seq 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.
- **Transformer (TF)** [226]: 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.
- **Latent Neural Quasi-Synchronous Context-Free Grammars (NQCFG)** [215]: 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.
- **Meta-Path Translation Model using CNN (MT-CNN)**: the proposed meta-path translation model using CNN [225] 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 $K_t = 50$ was used for inference.
- **Meta-Path Translation Model using Transformer (MT-TF)**: the proposed meta-path translation model using Transformer [226] 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**.

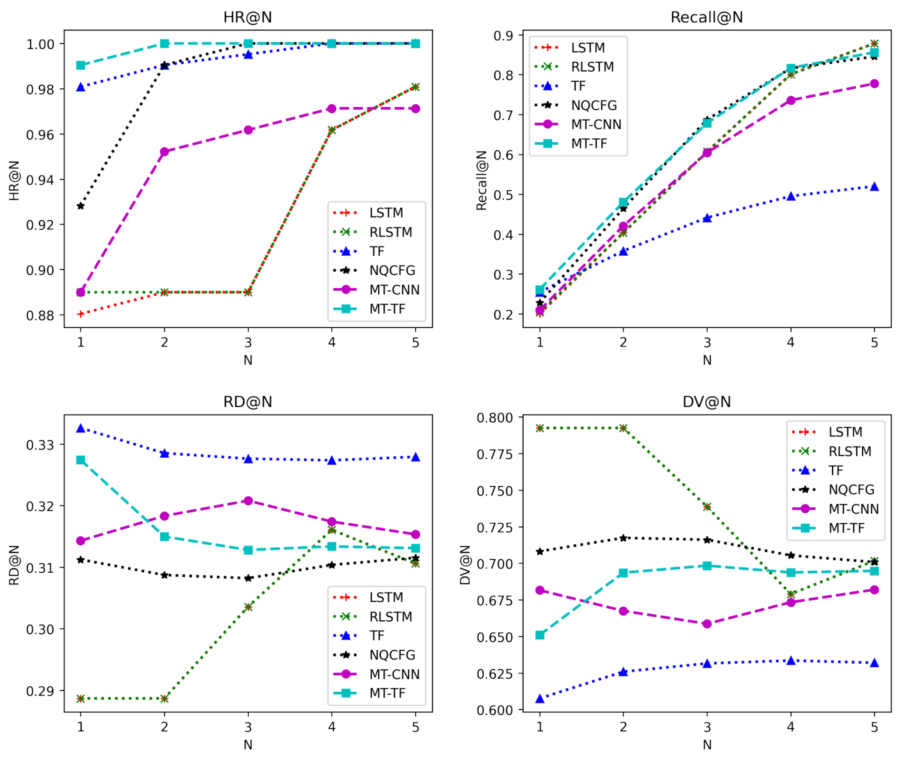
During the inference step of **LSTM**, **RLSTM** and **TF**, a beam search strategy [230] was adopted to generate top-N predictions. In this way, it is possible to compare the performances of predicting a set of comparable explainable meta-paths.

5.6.1 Comparison of the proposed approach and the baselines

Figures 5.8a and 5.8b 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 5.5 and Table 5.6 respectively. 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 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 the proposed models when N increased. **NQCFG** performed similarly to **TF** except when $N = 1$. Considering the proposed 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 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 CNN for extracting latent features in the proposed model for **MovieLens** dataset.



(a) MovieLens



(b) Amazon

Figure 5.8: Comparison of HR@N, Recall@N, RD@N and DV@N of the proposed approaches and other baselines

Table 5.5: HR@N and Recall@N improvements over the baselines on **MovieLens** dataset. The improvements that are statistically 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 5.6: HR@N and Recall@N improvements over the baselines on **Amazon** dataset. The improvements that are statistically 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

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 it performed worse than the other models including both of the proposed 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 the proposed models, **MT-TF** performed better than **MT-CNN** in terms of both HR@N and Recall@N. This suggests that using a Transformer model 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, the proposed model **MT-TF** predicted the target meta-paths more effectively than **MT-CNN** and the other baselines. **MT-TF** outperformed **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 CNN and Transformer.

5.6.2 Readability and Diversity Analysis

The results of RD@N and DV@N on **MovieLens** dataset and **Amazon** dataset are shown in Figures 5.8a and 5.8b 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 5.7 and Table 5.8 respectively. 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 the proposed models with the baselines, Table 5.7 shows the significant RD@N and DV@N improvements of **MT-CNN** and **MT-TF** over the baselines. From this table, **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 the proposed 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 their results on **MovieLens** dataset. It shows that they both 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 the proposed models in terms of RD@N but outperformed them in terms of DV@N. Table 5.8 shows the significant RD@N and DV@N improvements of **MT-CNN** and **MT-TF** over the baselines. From this table, **MT-CNN** significantly outperformed **LSTM** and **RLSTM** in terms of RD@N for every N except $N = 4$. It also significantly outperformed **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. It also significantly outperformed **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 the proposed 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. This result corresponds with the result on **MovieLens** dataset.

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**.

Table 5.7: RD@N and DV@N improvements over the baselines on **MovieLens** dataset. The improvements that are statistically 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 5.8: RD@N and DV@N improvements over the baselines on **Amazon** dataset. The improvements that are statistically significant are in bold. The value indicated with a star (*) is a positive and statistically significant value.

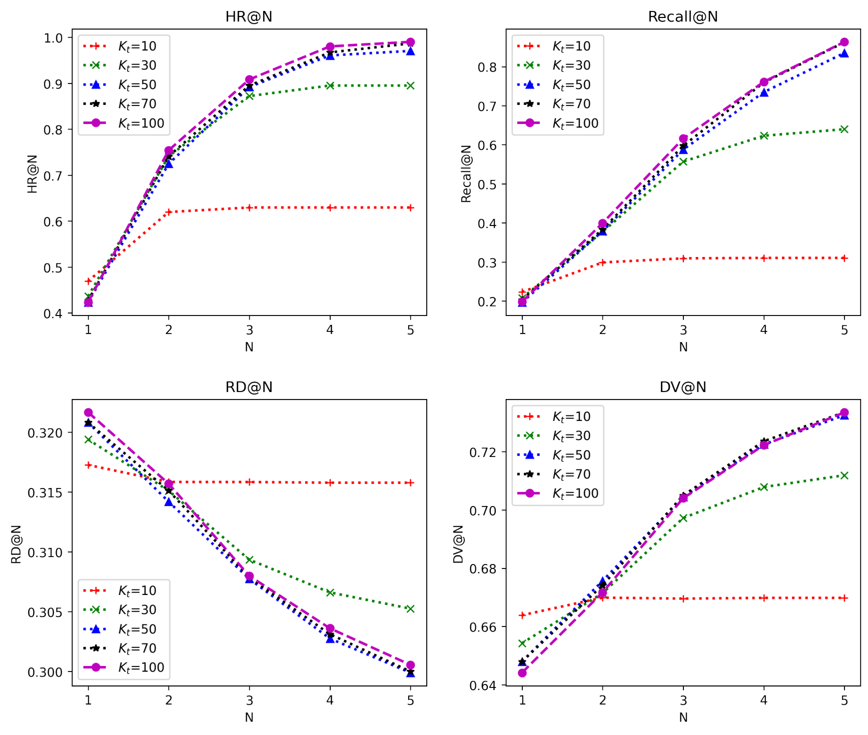
Baseline	MT-CNN									
	Δ RD@1	Δ RD@2	Δ RD@3	Δ RD@4	Δ RD@5	Δ DV@1	Δ DV@2	Δ DV@3	Δ DV@4	Δ DV@5
LSTM	**0.0256***	**0.0296***	**0.0172***	0.0013	**0.0047***	**-0.1109**	**-0.1250**	**-0.0800**	-0.0056	**-0.0197**
RLSTM	**0.0256***	**0.0296***	**0.0172***	0.0013	**0.0047***	**-0.1109**	**-0.1250**	**-0.0800**	-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

5.6.3 Hyperparameter Analysis

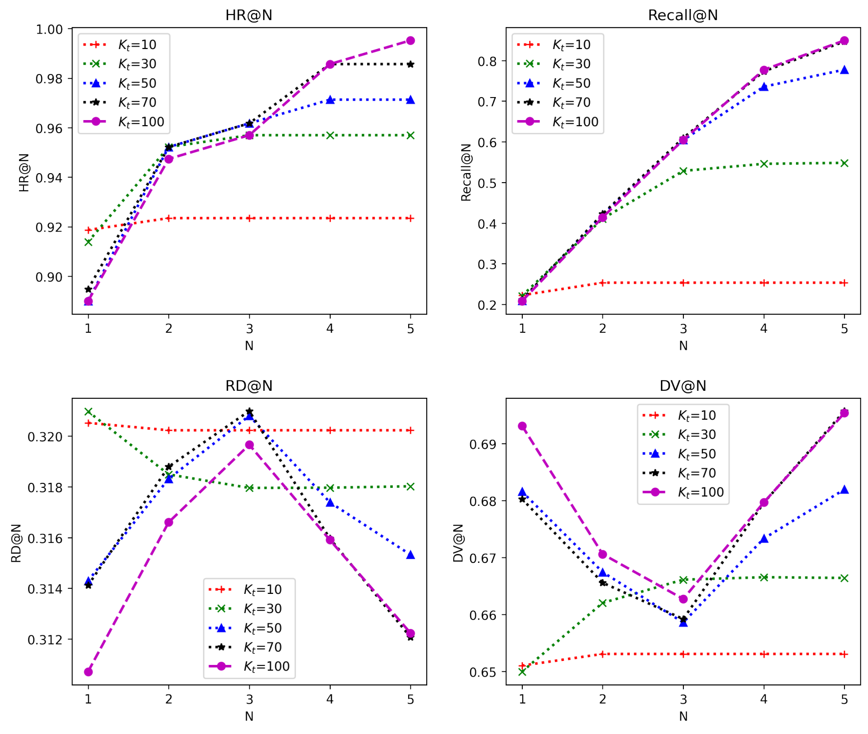
5.6.3.1 Effect of the number of sampled target trees (K)

In this section, the effect of the parameter K_t (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_t on both datasets are shown in Figure 5.9. From this figure, the higher values of K_t generally lead to better HR@N and Recall@N on both datasets, as both metrics increase with an increase in K_t for every value of N. However, when comparing $K_t = 50$, 70, and 100, the differences in HR@N and Recall@N are almost negligible. This suggests that using $K_t = 50$ 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_t increases. Conversely, DV@N of these meta-paths increases with an increase in K. On **Amazon** dataset, RD@N generally decreases as K_t 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_t, except for DV@3.

The results obtained from **MT-TF** on both datasets are presented in Figure 5.10. This figure illustrates that as the value of K_t increases, **MT-TF** predicts target meta-paths with higher HR@N and Recall@N on both datasets. Regarding Readability, on the **MovieLens** dataset, RD@N increases with the increment of K_t, whereas DV@N decreases. Conversely, on the **Amazon** dataset, the results are opposite to those of the **MovieLens** dataset. RD@N decreases with the increase in K_t, while DV@N increases. Overall, for both **MT-CNN** and **MT-TF**, HR@N and Recall@N improve with the increase in K_t. However, using larger values of K_t is computationally expensive, and therefore, using $K_t = 50$ is sufficient as it yields similar results to $K_t = 100$ in most cases. On the other hand, the trends observed in RD@N and DV@N vary among different models and datasets. Generally, using $K_t = 50$ achieves moderate results balancing between RD@N and DV@N.

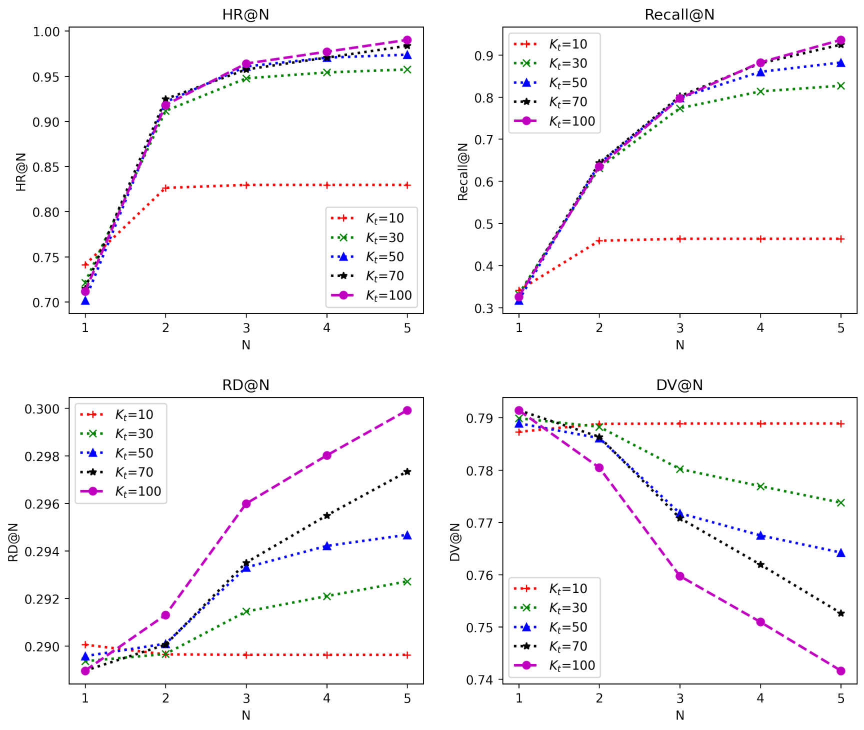


(a) MovieLens

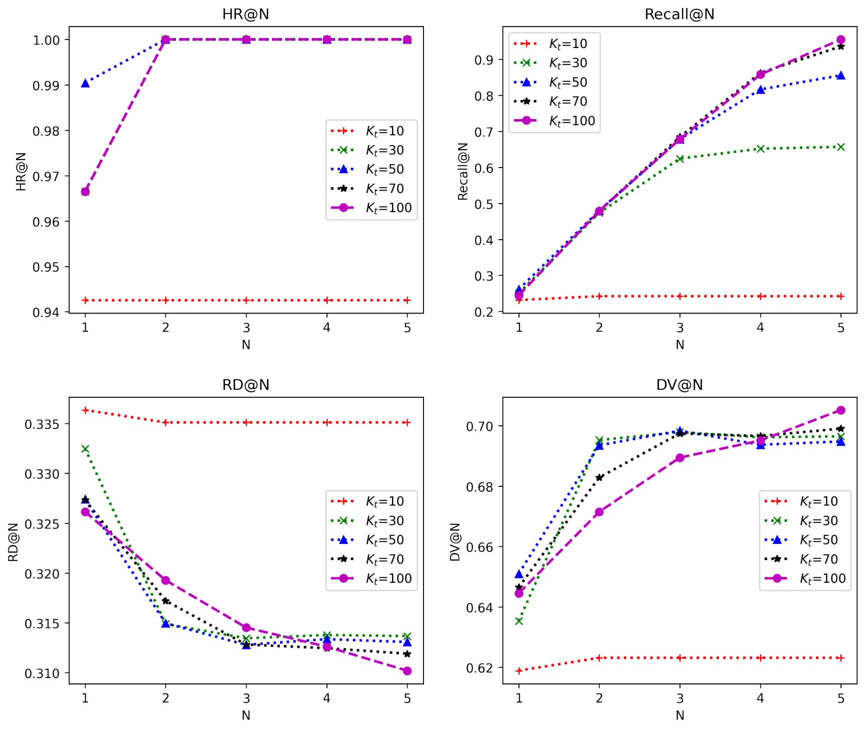


(b) Amazon

Figure 5.9: Comparison of HR@N and Recall@N when using different values of the number of sampled target trees from the decode (K_t) for sampling the target trees in the proposed meta-path translation model using CNN for latent feature extraction (**MT-CNN**) on (a) **MovieLens** dataset and (b) **Amazon** dataset



(a) MovieLens



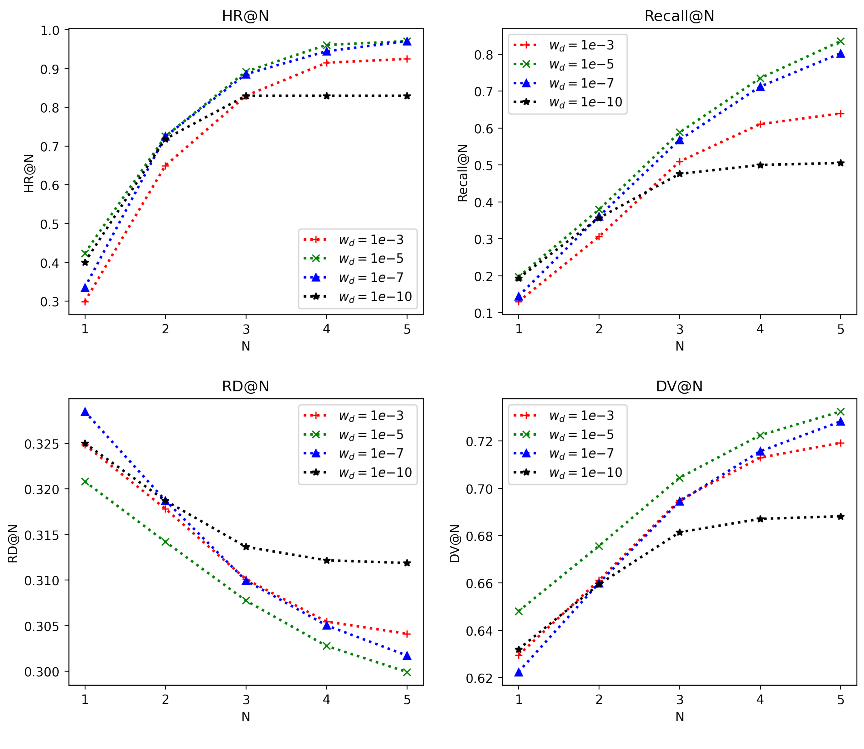
(b) Amazon

Figure 5.10: Comparison of HR@N and Recall@N when using different values of the number of sampled target trees from the decode (K_t) for sampling the target trees in the proposed meta-path translation model using Transformer for latent feature extraction (**MT-TF**) on (a) **MovieLens** dataset and (b) **Amazon** dataset

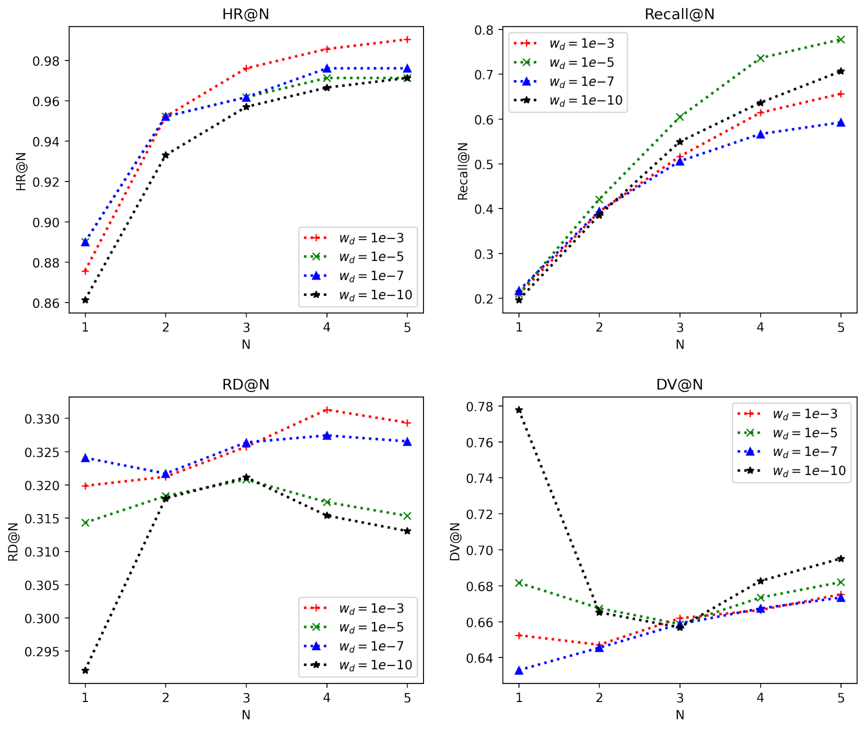
5.6.3.2 Effect of weight decay (w_d)

The influence of weight decay (w_d) on training the proposed models was also assessed by varying it among 10^{-3}, 10^{-5}, 10^{-7}, and 10^{-10}. Figures 5.11a and 5.11b display the results of weight decay variations in **MT-CNN** for **MovieLens** and **Amazon** datasets, respectively. On **MovieLens** dataset, **MT-CNN** using $w_d = 10^{-5}$ achieved the highest HR@N and Recall@N. On **Amazon** dataset, **MT-CNN** using $w_d = 10^{-5}$ achieved only the highest Recall@N. In terms of HR@N, **MT-CNN** using $w_d = 10^{-3}$ generally performed better than the others. However, the HR@N difference between using $w_d = 10^{-3}$ and using $w_d = 10^{-5}$ is less than the Recall@N difference between using $w_d = 10^{-3}$ and using $w_d = 10^{-5}$. Thus, $w_d = 10^{-5}$ was selected for comparison with the baselines on **Amazon** dataset. In terms of RD@N, on **MovieLens** dataset, **MT-CNN** using $w_d = 10^{-10}$ generally performed better than the others. However, it performed worse than the others in terms of DV@N. On the other hand, **MT-CNN** using $w_d = 10^{-5}$ performed better than the others in terms of DV@N but performed worse than the others 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 5.12a and 5.12b 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. In summary, the results show that extreme weight decay values can lead to decreased accuracy. On **MovieLens** dataset, lower weight decay improved performance, while on **Amazon** dataset, higher weight decay, especially for **MT-TF**, enhanced performance. It also can be seen that the RD@N and DV@N performances of each model are inversely proportional. If the model performed well in terms of RD@N, it would perform poorly in terms of DV@N, and vice versa.

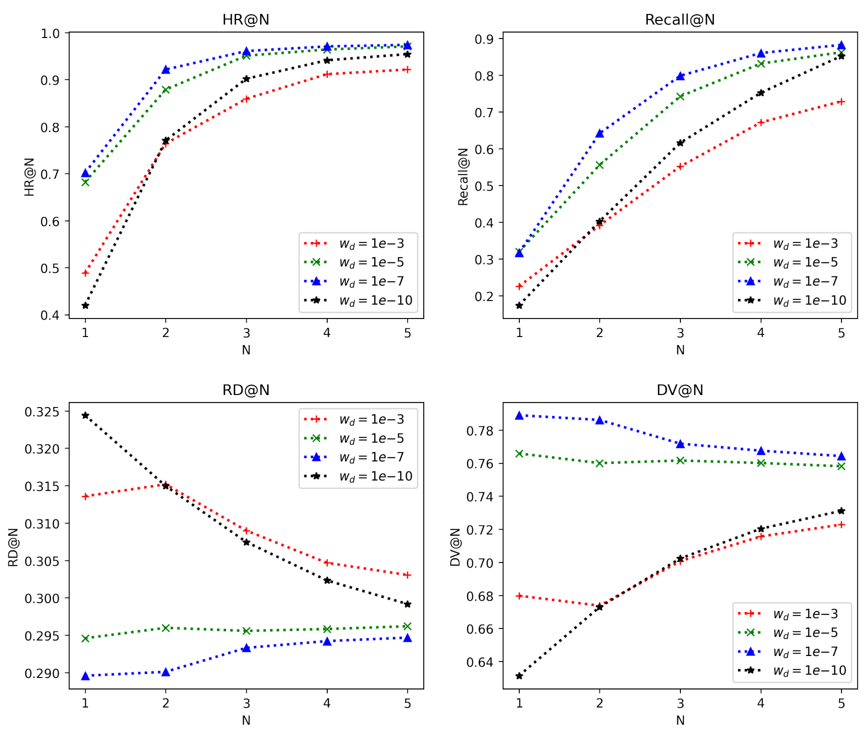


(a) MovieLens

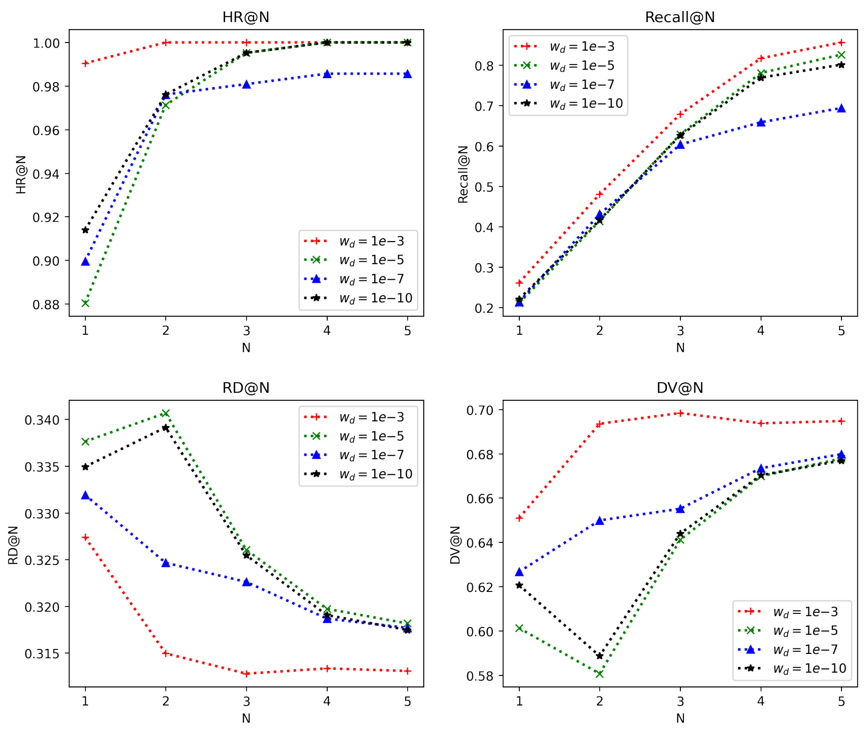


(b) Amazon

Figure 5.11: Comparison of HR@N and Recall@N when using different values of weight decay (w_d) for training the proposed meta-path translation model using CNN for latent feature extraction (**MT-CNN**) on (a) **MovieLens** dataset and (b) **Amazon** dataset



(a) MovieLens



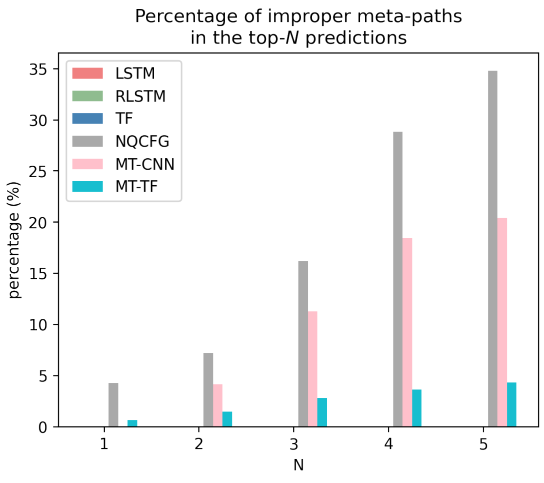
(b) Amazon

Figure 5.12: Comparison of HR@N and Recall@N when using differentvalues of weight decay (w_d) for training the proposed meta-path translation model using Transformer for latent feature extraction (**MT-CNN**) on (a) **MovieLens** dataset and (b) **Amazon** dataset

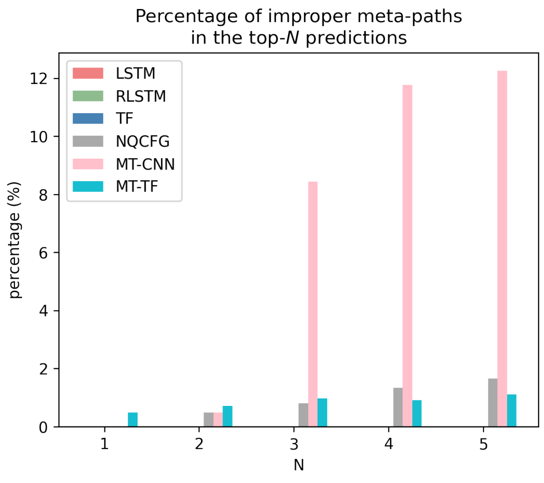
5.6.4 Error Analysis

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 $UN_1N_2, ..., N_lP$, $R_{UN_1}, R_{N_1,N_2}, ..., R_{N_l,P} \in \mathbb{R}$. 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 5.13. From this figure, there is no improper met-path predicted by **LSTM**, **RLSTM**, and **TF** on both **MovieLens** and **Amazon** datasets for every N. This demonstrates their effectiveness in modeling dependencies between the first and the last node types and also two consecutive node types in the target meta-paths. For **MovieLens**, the proposed 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 5.6.1.



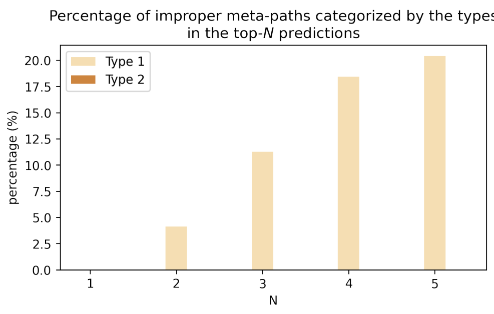
(a) MovieLens



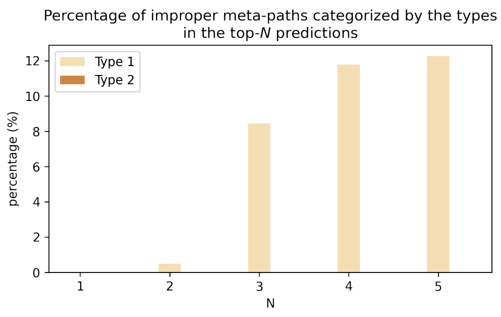
(b) Amazon

Figure 5.13: Comparisons of the percentages of improper meta-paths in the top-N predictions obtained from the proposed approaches and other baselines. An improper meta-path refers to a predicted target meta-path that does not start with the user node type and ends with the item node type or contains invalid relation types defined in the HIN schema.

The improper meta-paths generated from both **MT-CNN** and **MT-TF** were further examined to identify which type of improper meta-paths were predicted the most. Figures 5.14 and 5.15b depict comparisons of the percentages of improper meta-paths categorized by their types in the top-N recommendations that were predicted by **MT-CNN** and **MT-TF** respectively. These types are (1) improper meta-paths that do not start with the user node type U and end with the item node type P and (2) improper meta-paths that consist of non-existing relation type(s) which are referred to as Type 1 and Type 2 respectively. From Figure 5.14, the improper meta-paths predicted by **MT-CNN** are only Type 1 on both datasets. Meanwhile, Figure 5.15 shows that the improper meta-paths predicted by **MT-TF** are both Type 1 and Type 2. However, the percentages of Type-2 improper meta-paths are less than 0.5% for every N on both datasets. The majority of these improper meta-paths obtained **MT-TF** are therefore Type 1. From these results, the proposed models both suffered from predicting Type-1 improper meta-paths especially **MT-CNN**. One reason that **MT-TF** predicted fewer Type-1 improper meta-paths could be that all node types from the beginning to the end of a sequence are attended to when learning latent features at each and every layer of the Transformer. Relationships between all nodes including the first and the last nodes can be learned at every layer. On the contrary, only relationships between node types within a convolution kernel can be captured at each layer in CNN. This allows for a relationship between the first and the last node types to be captured at the last convolutional layer. Therefore, a relationship between the first node type and the last node type can be captured more easily in Transformer than in CNN.

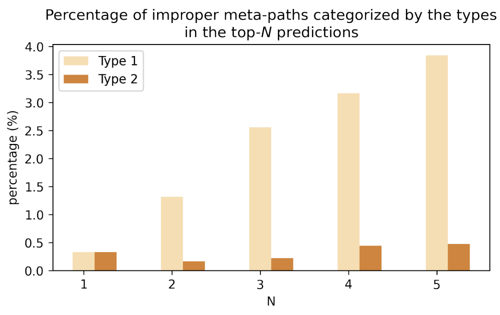


(a) MovieLens

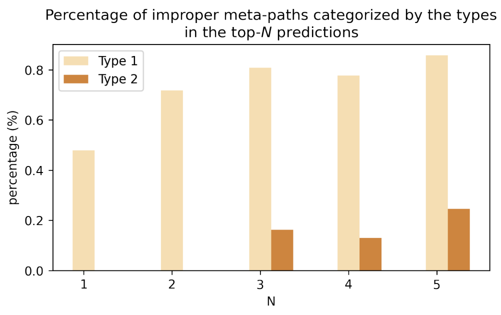


(b) Amazon

Figure 5.14: Comparisons of the percentages of improper meta-paths in the top-N recommendations that were predicted by **MT-CNN** categorized by types, i.e., improper meta-paths that do not start with the user node type U and end with the item node type P (Type 1) and improper meta-paths that consist of non-existing relation type(s) (Type 2) on (a) **MovieLens** dataset and (b) **Amazon** dataset



(a) MovieLens



(b) Amazon

Figure 5.15: Comparisons of the percentages of improper meta-paths in the top-N recommendations that were predicted by **MT-TF** categorized by types, i.e., improper meta-paths that do not start with the user node type U and end with the item node type P (Type 1) and improper meta-paths that consist of non-existing relation type(s) (Type 2) on (a) **MovieLens** dataset and (b) **Amazon** dataset

5.6.5 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 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.

5.7 Summary

Meta-path based recommendations provide an intuitive approach to explain the underlying rationale behind the recommendations. These explanations are derived from the meanings of the meta-paths used in generating the recommendations. However, explanations may become difficult to understand when recommendations rely on lengthy and complex meta-paths. To address this issue, this chapter places its focus on enhancing the explainability of meta-path based recommendations. This objective aligns with the third research question, which aims to further improve the explainability of explainable visually-aware recommender systems based on HINs. To achieve this goal, the concept of meta-path explainability was introduced. Then, based on this concept, a method was proposed to identify comparable explainable meta-paths. These meta-paths were selected based on their ability to deliver similar recommendation performance while offering higher levels of explainability. They were used to provide a clearer understanding of the underlying rationale behind the meta-path based recommendations. To facilitate the identification of comparable explainable meta-paths, the concept of meta-path grammar was introduced, allowing meta-paths to be constructed in a manner similar to sentences in human languages. Based on this grammar, a meta-path translation model was introduced. This model translated a meta-path to its comparable explainable meta-paths by leveraging both latent features extracted by CNN/Transformer and hierarchical features from the TreeLSTM model.

Two meta-path translation datasets were generated using real-world datasets. Extensive experiments on these datasets demonstrated that the proposed model outperformed the baselines in terms of HR@N and Recall@N for both datasets. The approach achieved a good trade-off between accuracy and readability/diversity, with the CNN model performing better in readability and the Transformer model excelling in diversity. Furthermore, the inclusion of latent feature embeddings and positional embeddings resulted in fewer improper target meta-paths compared to the traditional model without these embeddings.

Overall, the proposed method can successfully improve the explainability of meta-path based recommendations by providing alternative and more understandable explanations. As the proposed visually-aware recommender systems and explainable visually-aware recommender systems presented in previous chapters are meta-path based, their explainability can then be further enhanced by the proposed meta-path translation method. Thus, the objective of enhancing the explainability of visually-aware recommender systems is accomplished in this chapter.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Recommender systems, ubiquitous in various online platforms, play a pivotal role in helping users navigate the overwhelming sea of online content. Their significance lies in addressing the challenge of information overload and assisting users in discovering personalized and relevant items. However, the effectiveness of traditional recommender systems is constrained by the sparsity problem, particularly when user-item interactions are limited. To overcome this limitation, a key strategy involves incorporating additional information, with a growing emphasis on leveraging images to enhance recommendation accuracy. This gives rise to visually-aware recommender systems, a paradigm shift that considers not only user-item interactions but also the rich visual features encapsulated in item images. Despite their strides in improving recommendation accuracy, a critical challenge arises concerning the lack of explainability in visually-aware recommender systems.

Explainability ensures that users can comprehend the decision-making process of a recommender system, fostering transparency in how recommendations are generated. In scenarios where users depend on the system's suggestions, the capability to provide clear and understandable explanations becomes a fundamental aspect of ensuring user trust and satisfaction. To address this concern, explainable recommender systems have emerged. Among various approaches to developing explainable recommender systems, leveraging HINs in explainable recommender systems has become a ubiquitous practice. However, a notable gap exists in current approaches, as they predominantly focus on semantic information within HINs, neglecting the potential integration of visual information. Bridging this gap is crucial for the development of explainable visually-aware recommender systems, ensuring not only accuracy but also transparency in the recommendations provided to users.

This work develops explainable visually-aware recommender systems using HINs that produce accurate and explainable recommendations personalized to users' visual preferences. To do so, **visually-augmented HINs** in which image features are incorporated as visual factor nodes connected with other nodes via visual relations was first proposed. Five different types of image feature extraction methods, i.e., SIFT, SURF, ORB, Colo Histogram, and the pre-trained CNN model, were adopted to generate these visual factor nodes and visual relations. Moreover, a user representation learning approach using visually-augmented HINs was proposed. This approach uses a **probabilistic meta-path** to form a hybrid context that considers both semantic and visual factors to profile users. Extensive experiments were conducted to examine the possibility of using these HINs to learn visually-aware recommendations. The user representations generated from the proposed approach were applied to user-based CF-KNN recommender systems on two real-world datasets, HetRec2011-MovieLens-2K and Amazon-Clothing datasets. The proposed approaches were compared with the baseline models using metapath2vec++ [85]. The results show that visually-augmented HINs and probabilistic meta-paths can be used to improve recommendation accuracy. The results also depict that, among different types of image features, Color Histogram and CNN features are more effective in capturing users' visual preferences on HetRec2011-MovieLens-2K dataset compared to the other features. Meanwhile, on Amazon-Clothing dataset, SURF performed better than the others. This suggests that texture and shape features are more effective in recognizing users' preferences for clothing items. Also, the visually-augmented HINs based on these two datasets were also applied with the state-of-the-art HIN-based recommendation model called Graph Attention Network [5] to evaluate the effectiveness of these augmented HINs. The results indicate that using visually-augmented HINs can effectively be applied to the state-of-the-art HIN-based recommendation model. They can improve the accuracy performance of the Graph Attention Network model since augmenting HINs with visual factor nodes and visual relations can increase the chance of discovering more relationships apart from those in the original HINs. However, in the case that there are already a large number of nodes and relations in the original HIN, augmenting visual factor nodes and visual relations may affect the overall effectiveness of the Graph Attention Network model.

Next, a **scalable and explainable visually-aware recommender system framework (SEV-RS)** was proposed. This framework utilizes multi-hop relations in visually-augmented HINs to predict explainable visually-aware recommendations. In this framework, a scalable method to extract meta-path based features from visually-augmented HINs was proposed. This method can efficiently leverage a set of meta-paths for the better use of multi-hop relations. To achieve explainability, a novel concept of meta-path based explainability was introduced. This meta-path based explainability allows one to quantify the explainability score of a user-item pair based on a set of meta-paths using extracted meta-path features. Lastly, to generate explainable recommendations, a shallow BPR-based recommendation model was proposed. This model jointly leverages the proposed meta-path features and the meta-path based explainability to learn explainable visually-aware recommendations. SEV-RS was evaluated in the Top-N recommendation task based on three evaluation metrics, i.e., Accuracy, Explainability, and Scalability. Extensive experiments were conducted on HetRec2011-MovieLens-2K dataset and Amazon-Clothing dataset and one syntactic dataset. The results show that SEV-RS can produce more accurate recommendations according to the higher Precision and Recall compared with the baselines. Compared to the Graph Attention Network model, SEV-RS can leverage visual information in visually-augmented HINs more effectively than the Graph Attention Network model. Regarding Explainability, the results show that SEV-RS can generate more explainable recommendations with higher Explainability Precision and Explainability Recall. This indicates that the proposed approach does not only recommend items of users' interests but also takes the explainability of items into consideration. As for Scalability, experiments were conducted on a synthetic dataset consisting of four sub-datasets with different scales. The computational time of each model was compared. The results show that SEV-RS can achieve similar scalability performances as the BPR-MF model which is a shallow model. It also required much less computational time compared to the Graph Attention Network model for large-scale sub-datasets. The results also indicate that the additional computational time required for executing the explainability part in SEV-RS is trivial and thus this approach is scalable. Lastly, the proposed scalable meta-path feature extraction method and the straight-forwarding non-scalable method were compared in terms of performance. The results show that using the proposed scalable method achieved similar Accuracy but cost significantly less computational time than using the straight-forwarding method.

The aforementioned results have shown that explainable visually-aware recommendations can be achieved by using visually-augmented HINs and SEV-RS. The explainability of these recommendations is based solely on meta-paths adopted in the framework. As a result, if the adopted meta-paths are complicated and difficult to understand, they will affect the explainability of the recommendations based on them. To further improve the explainability of not only SEV-RS but also any meta-path based recommender systems, this thesis proposed a method that can translate a complicated meta-path to a group of meta-paths that are more comprehensible but perform similarly in terms of recommendation accuracy. First, the definition of **meta-path explainability** was introduced. This definition is three meta-path quality measurements, i.e., meta-path readability, meta-path credibility, and meta-path diversity. Based on this definition, a method to find **comparable explainable meta-paths** of a given meta-path was proposed. These comparable explainable meta-paths represent those meta-paths that are relatively more explainable than a given one but yield similar recommendation accuracy. To find comparable explainable meta-paths of any meta-path, a **meta-path translation model** was proposed. The meta-path grammar inspired by QCFG was first introduced. Based on this 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 and hierarchical features extracted by TreeLSTM simultaneously. Two meta-path translation datasets were generated based on HetRec2011-MovieLens-2K dataset and Amazon-Clothing dataset. In these generated 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 were used to learn how to explain recommendations of the 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 the 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, the proposed 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 model using CNN performed better in terms of Readability. On the other hand, the proposed model using Transformer produced the target meta-paths with higher Diversity. Both models 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, the proposed 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.

In summary, this thesis advances the development of explainable visually-aware recommender systems through the utilization of HINs. These recommender systems are designed to generate recommendations that are both accurate and explainable, personalized to users' visual preferences. The implications of these advancements extend to practical applications. The proposed approaches pave the way for developing recommender systems that cater to users' visual preferences and provide explanations rooted in interpretable meta-paths. The application of these approaches can contribute to a more engaging and personalized user experience in various domains such as e-commerce, and entertainment. Furthermore, the meta-path translation model, introduced to enhance the explainability of meta-path based recommender systems, presents an opportunity for refining and enriching the transparency of recommendations. The capability to translate complex meta-paths into simpler and more understandable ones offers a strategic approach to increase user trust and comprehension of recommendations in real-world scenarios. Furthermore, beyond their practical implications, the proposed approaches hold theoretical potential for broader application. They can be adapted and generalized to improve other instances of explainable AI applications, especially those utilizing graphs or networks. Also, as deep-learning methods typically suffer from a lack of explainability, these approaches can serve as alternatives to these methods. Overall, the work undertaken in this thesis contributes to the advancement of visually-aware recommender systems that are accurate, explainable, and scalable.

6.2 Contributions

The contributions of this thesis are summarized as follows:

- This thesis contributes to the advancement of visually-aware recommender systems by integrating visual information into a HIN, constructing visually-augmented HINs with various visual feature extraction methods. It introduces probabilistic meta-paths for forming hybrid contexts, combining semantic and visual information to learn user latent representations. These representations were applied with a user-based CF-KNN model for building effective visually-aware recommender systems and validated through extensive experiments on real-world datasets.

- This thesis contributes to the development of an explainable visually-aware recommender system framework based on HINs. The proposed framework utilizes a visually-augmented

HIN, a scalable meta-path feature extraction method, and an explainable recommendation generation method. Extensive experiments were conducted to evaluate the proposed framework on accuracy, explainability, and scalability, demonstrating its effectiveness in explainable visually-aware recommendations.

- This thesis contributes to the improvement of the explainability of meta-path based recommender systems by introducing the novel meta-path translation task. Addressing challenges posed by complex meta-paths in recommendations, this task aims to translate meta-paths into simpler, more interpretable alternatives, providing improved comprehension. This thesis defines meta-path explainability, proposes a method to find comparable explainable meta-paths, introduces meta-path grammar, and presents a meta-path translation model. Experiments conducted on novel datasets demonstrate the effectiveness of the proposed model.

6.3 Limitations

First, the limitations of visually-augmented HINs and probabilistic meta-paths are discussed. In visually-augmented HINs, visual information is incorporated through the utilization of visual factor nodes. These nodes are representatives of significant features derived from clustering analysis performed on all extracted image features. One noteworthy limitation pertains to the lack of semantic explicitness exhibited by these representatives. Specifically, it remains ambiguous which specific characteristics or appearances each visual factor node signifies. For instance, while one visual factor node may denote a light-color characteristic prevalent in images, another node might indicate a dark-color characteristic. Identifying the corresponding characteristics associated with each visual factor node may further enhance the explainability of visually-aware recommendations using visually-augmented HINs. Considering probabilistic meta-paths, the assigned probabilities must be pre-defined in order to model hybrid users' profiles. Determining these values to effectively utilize such a meta-path relies solely on experts or prior knowledge. These pre-defined probabilities are also constant. This means that the same probabilities are applied to every user. However, in reality, different users may be influenced by different degrees of visual factors. Therefore, the ratio between semantic factors and visual factors for modeling hybrid users' profiles might need to be varied individually. SEV-RS also has some limitations. One of them is the selection of meta-paths used for extracting meta-path features. The determination of these meta-paths requires prior knowledge or expertise to effectively identify suitable meta-paths. This prerequisite poses a challenge as it relies on manual human intervention and decision-making.

As for the proposed meta-path translation model, identifying comparable explainable meta-paths can be computationally expensive since the meta-path based recommender systems based on multiple meta-paths are required. It is important to recognize that different meta-path based recommendation approaches have varying computational complexities. In case the chosen approach is computationally expensive, it will cost the process of identifying comparable explainable meta-paths to be time-consuming as well. These can be considered as a limitation of the proposed approach and can be addressed in future work.

6.4 Future work

Future work will involve the exploration of more advanced recommendation models, which will be applied with visually-augmented HINs. Through the exploration of different models, a comprehensive understanding of how visually-augmented HINs can enhance various recommendation approaches will be gained. Furthermore, additional analysis of visual factor nodes will be conducted with the objective of annotating their semantic meanings. This analysis will facilitate users in comprehending the specific characteristics or appearances associated with each visual factor node. Additionally, a broader range of visual feature extraction methods, as well as techniques for feature fusion and dimension reduction, will be explored to capture diverse types of image features. Such an approach holds the potential to provide more comprehensive representations of image features and enhance the modeling of users' visual preferences. To enhance flexibility, personalized probabilistic meta-paths will be incorporated to cater to individual user profiles, enabling the capture of hybrid preferences. This adaptive approach will enable the utilization of more appropriate probabilistic meta-paths tailored to the specific needs of individual users.

Future research efforts will also focus on adopting alternative recommendation models and applying them with meta-path features and meta-path based explainability. This extended study will provide a more comprehensive understanding of how meta-path features can benefit other recommendation models, as well as how meta-path based explainability can enhance the interpretability of other recommendation models. Furthermore, since the selection of appropriate meta-paths for generating accurate and explainable recommendations poses a significant challenge, future work will concentrate on the systematic selection or validation of meta-paths. The incorporation of such a meta-path selection mechanism will not only facilitate the development of meta-path based recommender systems but also enhance the overall explainability of these systems. Moreover, considering large-scale real-world datasets to evaluate the proposed scalable and explainable visually-aware recommender system framework would be beneficial. Applying it to such datasets will allow for a comprehensive assessment of its effectiveness in terms of both recommendation accuracy and computational scalability.

Future work will also involve exploring advanced machine learning approaches, such as contrastive learning and self-supervised learning, to enhance the process of learning meta-path translation. In the experiments in this thesis, two neural network architectures (i.e., CNN and Transformer) were considered when building meta-path translation models. To further examine the effectiveness of the proposed approach and improve the performance of meta-path translation models, it is worth exploring other neural network architectures for extracting latent features from a meta-path. Additionally, considering assigning different relation weights in HINs to examine the performance of the proposed meta-path translation approach could be beneficial. With distinct weights assigned to different types of relations, meta-path credibility can be factored into the translation of meta-paths. This will allow for further studies on the influence of meta-path credibility on the meta-path translation process. In terms of experiments, considering other real-world datasets to generate more meta-path translation datasets for explaining meta-path based recommendations would provide a more comprehensive study of meta-path translation in different domains. Also, since the task of meta-path translation is newly introduced, generating more datasets will provide more options for other researchers interested in exploring this task. As generating datasets for training meta-path translation models can be highly time-consuming, the future work will address this concern and improve the process of generating meta-path translation datasets to enable more scalable applications. Lastly, the potential adoption of Large Language Models (LLMs) such as BERT or GPT models will be investigated. It will involve harnessing the immense linguistic knowledge embedded in pre-trained LLMs and combining it with HINs to enhance the explainability of recommendations. This approach will not only benefit meta-path based recommendations but also HIN-based recommendations in general.

References

[1] “Leather jackets for men.” https://www.amazon.co.uk/. Accessed: 2023-10-09.

[2] “Recommended topic - mental health.” https://patient.info/. Accessed: 2023-10-09.

[3] “Recommended for you.” https://www.udacity.com/. Accessed: 2023-10-09.

[4] “Tailored fit pure cotton herringbone shirt.” https://www.marksandspencer.com/. Accessed: 2023-10-09.

[5] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, “Kgat: Knowledge graph attention network for recommendation,” in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, p. 950–958, 2019.

[6] S. Zhang, L. Yao, A. Sun, and Y. Tay, “Deep learning based recommender system: A survey and new perspectives,” *ACM Computing Surveys*, vol. 52, Feb. 2019.

[7] J. Lu, D. Wu, M. Mao, W. Wang, and G. Zhang, “Recommender system application developments: A survey,” *Decision Support Systems*, vol. 74, pp. 12–32, 2015.

[8] D. Jannach and M. Jugovac, “Measuring the business value of recommender systems,” *ACM Trans. Manage. Inf. Syst.*, vol. 10, dec 2019.

[9] R. Mu, “A survey of recommender systems based on deep learning,” *IEEE Access*, vol. 6, pp. 69009–69022, 2018.

[10] Y. Zhang and X. Chen, “Explainable recommendation: A survey and new perspectives,” *Foundations and Trends® in Information Retrieval*, vol. 14, no. 1, p. 1–101, 2020.

[11] R. F. Doh, C. Zhou, J. K. Arthur, I. Tawiah, and B. Doh, “A systematic review of deep knowledge graph-based recommender systems, with focus on explainable embeddings,” *Data*, vol. 7, 2022.

[12] J. Bobadilla, F. Ortega, A. Hernando, and A. Gutiérrez, "Recommender systems survey," *Knowledge-Based Systems*, vol. 46, pp. 109–132, 2013.

[13] Q. Guo, F. Zhuang, C. Qin, H. Zhu, X. Xie, H. Xiong, and Q. He, "A survey on knowledge graph-based recommender systems," *IEEE Transactions on Knowledge & Data Engineering*, 2020.

[14] G. Adomavicius and A. Tuzhilin, "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*, vol. 17, p. 734–749, June 2005.

[15] S. Wang, L. Hu, Y. Wang, X. He, Q. Z. Sheng, M. A. Orgun, L. Cao, F. Ricci, and P. S. Yu, "Graph learning based recommender systems: A review," *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI-21)*, 2021.

[16] Y. Deldjoo, F. Nazary, A. Ramisa, J. Mcauley, G. Pellegrini, A. Bellogin, and T. Di Noia, "A review of modern fashion recommender systems," 2022.

[17] Q. You, S. Bhatia, and J. Luo, "A picture tells a thousand words—about you! user interest profiling from user generated visual content," *Signal Processing*, vol. 124, pp. 45–53, 2016. Big Data Meets Multimedia Analytics.

[18] A. Gupta and R. Jain, "Visual information retrieval," *Communications of the ACM*, vol. 40, no. 5, pp. 70–79, 1997.

[19] Y. Liu, D. Zhang, G. Lu, and W.-Y. Ma, "A survey of content-based image retrieval with high-level semantics," *Pattern recognition*, vol. 40, no. 1, pp. 262–282, 2007.

[20] A. Latif, A. Rasheed, U. Sajid, J. Ahmed, N. Ali, N. I. Ratyal, B. Zafar, S. H. Dar, M. Sajid, and T. Khalil, "Content-based image retrieval and feature extraction: A comprehensive review," *Mathematical Problems in Engineering*, vol. 2019, 2019.

[21] B. Manjunath and W. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837–842, 1996.

[22] N. Kingsbury and J. Magarey, "Wavelet transforms in image processing," 1998.

[23] Y. Mingqiang, K. Kidiyo, R. Joseph, *et al.*, "A survey of shape feature extraction techniques," *Pattern recognition*, vol. 15, no. 7, pp. 43–90, 2008.

[24] Y. Li, S. Wang, Q. Tian, and X. Ding, "A survey of recent advances in visual feature detection," *Neurocomputing*, vol. 149, pp. 736–751, 2015.

[25] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, pp. 1150–1157 vol.2, 1999.

[26] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *Proceedings of the 9th European Conference on Computer Vision*, vol. 3951, pp. 404–417, 07 2006.

[27] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to sift or surf," *2011 International Conference on Computer Vision*, 2011.

[28] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, *et al.*, "Recent advances in convolutional neural networks," *Pattern recognition*, vol. 77, pp. 354–377, 2018.

[29] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[30] W. Rawat and Z. Wang, "Deep convolutional neural networks for image classification: A comprehensive review," *Neural Computation*, vol. 29, no. 9, pp. 2352–2449, 2017.

[31] J. Wang, Y. Yang, J. Mao, Z. Huang, C. Huang, and W. Xu, "Cnn-rnn: A unified framework for multi-label image classification," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[32] B. H. van der Velden, H. J. Kuijf, K. G. Gilhuijs, and M. A. Viergever, "Explainable artificial intelligence (xai) in deep learning-based medical image analysis," *Medical Image Analysis*, vol. 79, p. 102470, 2022.

[33] I. Lenz, R. A. Knepper, and A. Saxena, "Deepmpc: Learning deep latent features for model predictive control.," in *Robotics: Science and Systems*, vol. 10, p. 25, Rome, Italy, 2015.

[34] X. Liu, R. Zhang, Z. Meng, R. Hong, and G. Liu, "On fusing the latent deep cnn feature for image classification," *World Wide Web*, vol. 22, pp. 423–436, 2019.

[35] Y. Wen, Z. Li, and Y. Qiao, "Latent factor guided convolutional neural networks for age-invariant face recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[36] K. Uehara, M. Murakawa, H. Nosato, and H. Sakanashi, "Multi-scale explainable feature learning for pathological image analysis using convolutional neural networks," in *2020 IEEE International Conference on Image Processing (ICIP)*, pp. 1931–1935, IEEE, 2020.

[37] R. He and J. McAuley, "VBPR: Visual bayesian personalized ranking from implicit feedback," in *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, p. 144–150, AAAI Press, 2016.

[38] W.-C. Kang, C. Fang, Z. Wang, and J. McAuley, "Visually-aware fashion recommendation and design with generative image models," *2017 IEEE International Conference on Data Mining (ICDM)*, pp. 207–216, 2017.

[39] R. He, C. Fang, Z. Wang, and J. McAuley, "Vista: A visually, socially, and temporally-aware model for artistic recommendation," in *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, (New York, NY, USA), p. 309–316, Association for Computing Machinery, 2016.

[40] X. Chen, H. Chen, H. Xu, Y. Zhang, Y. Cao, Z. Qin, and H. Zha, "Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation," *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 765–774, 2019.

[41] C. Yan and Q. Zhang, "Merging visual features and temporal dynamics in sequential recommendation," *Neurocomputing*, vol. 362, pp. 11 – 18, 2019.

[42] M. He, S. Zhang, and Q. Meng, "Learning to style-aware bayesian personalized ranking for visual recommendation," *IEEE Access*, vol. 7, pp. 14198–14205, 2019.

[43] Z. Cheng, X. Chang, L. Zhu, R. C. Kanjirathinkal, and M. Kankanhalli, "MMalfM: Explainable recommendation by leveraging reviews and images," *ACM Transactions on Information Systems*, vol. 37, no. 2, 2019.

[44] C. Packer, J. McAuley, and A. Ramisa, "Visually-aware personalized recommendation using interpretable image representations," 2018.

[45] M. Hou, L. Wu, E. Chen, Z. Li, V. W. Zheng, and Q. Liu, "Explainable fashion recommendation: A semantic attribute region guided approach," in *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2019-Augus, pp. 4681–4688, 2019.

[46] Y. Deldjoo, T. Di Noia, D. Malitesta, and F. A. Merra, "A study on the relative importance of convolutional neural networks in visually-aware recommender systems," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 3956–3962, 2021.

[47] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, and F. Herrera, "Explainable artificial intelligence (xai): Concepts, taxonomies, opportunities and challenges toward responsible ai," *Information Fusion*, vol. 58, pp. 82–115, 2020.

[48] S. Mohseni, N. Zarei, and E. D. Ragan, "A multidisciplinary survey and framework for design and evaluation of explainable ai systems," *ACM Transactions on Interactive Intelligent Systems*, 2021.

[49] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision-making and a "right to explanation"," *AI Magazine*, vol. 38, pp. 50–57, Oct. 2017.

[50] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Association for Computing Machinery, 2016.

[51] M. T. Ribeiro, S. Singh, and C. Guestrin, "Anchors: High-precision model-agnostic explanations," in *AAAI Conference on Artificial Intelligence*, 2018.

[52] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, (Red Hook, NY, USA), p. 4768–4777, Curran Associates Inc., 2017.

[53] P. Cortez and M. J. Embrechts, "Using sensitivity analysis and visualization techniques to open black box data mining models," *Information Sciences*, vol. 225, pp. 1–17, 2013.

[54] A. Henelius, K. Puolamäki, and A. Ukkonen, "Interpreting classifiers through attribute interactions in datasets," in *Proceedings of the 2017 ICML Workshop on Human Inter-*

pretability in Machine Learning (WHI 2017) (B. Kim, D. Malioutov, K. Varshney, and A. Weller, eds.), 2017.

[55] B. Kim, M. Wattenberg, J. Gilmer, C. J. Cai, J. Wexler, F. B. Viégas, and R. Sayres, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)," in *ICML*, 2018.

[56] A. Ghorbani, J. Wexler, J. Y. Zou, and B. Kim, "Towards automatic concept-based explanations," in *Advances in Neural Information Processing Systems* (H. Wallach, H. Larochelle, A. Beygelzimer, F. d Alché-Buc, E. Fox, and R. Garnett, eds.), vol. 32, Curran Associates, Inc., 2019.

[57] R. Agarwal, L. Melnick, N. Frosst, X. Zhang, B. Lengerich, R. Caruana, and G. E. Hinton, "Neural additive models: Interpretable machine learning with neural nets," in *Advances in Neural Information Processing Systems* (M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), vol. 34, pp. 4699–4711, Curran Associates, Inc., 2021.

[58] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," in *Workshop at International Conference on Learning Representations*, 2014.

[59] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Computer Vision – ECCV 2014* (D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, eds.), pp. 818–833, 2014.

[60] X. Chen, Y. Zhang, and J. Wen, "Measuring "why" in recommender systems: a comprehensive survey on the evaluation of explainable recommendation," *ArXiv*, vol. abs/2202.06466, 2022.

[61] H. Wang, F. Zhang, J. Wang, M. Zhao, W. Li, X. Xie, and M. Guo, "Ripplenet: Propagating user preferences on the knowledge graph for recommender systems," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, p. 417–426, 2018.

[62] Z. Yang and S. Dong, "HAGERec: Hierarchical Attention Graph Convolutional Network Incorporating Knowledge Graph for Explainable Recommendation," *Knowledge-Based Systems*, vol. 204, p. 106194, 2020.

[63] Y. Sun, Y. Yu, and J. Han, "Ranking-based clustering of heterogeneous information networks with star network schema," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, (New York, NY, USA), p. 797–806, Association for Computing Machinery, 2009.

[64] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable Reasoning over Knowledge Graphs for Recommendation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 5329–5336, 2019.

[65] Y. Xian, Z. Fu, S. Muthukrishnan, G. de Melo, and Y. Zhang, "Reinforcement knowledge graph reasoning for explainable recommendation," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, p. 285–294, 2019.

[66] I. Tiddi and S. Schlobach, "Knowledge graphs as tools for explainable machine learning: A survey," *Artificial Intelligence*, vol. 302, p. 103627, 2022.

[67] K. Zhao, X. Wang, Y. Zhang, L. Zhao, Z. Liu, C. Xing, and X. Xie, "Leveraging demonstrations for reinforcement recommendation reasoning over knowledge graphs," in *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, (New York, NY, USA), p. 239–248, Association for Computing Machinery, 2020.

[68] X. Wang, K. Liu, D. Wang, L. Wu, Y. Fu, and X. Xie, "Multi-level recommendation reasoning over knowledge graphs with reinforcement learning," in *Proceedings of the ACM Web Conference 2022*, WWW '22, (New York, NY, USA), p. 2098–2108, Association for Computing Machinery, 2022.

[69] W. Ma, W. Jin, M. Zhang, C. Wang, Y. Cao, Y. Liu, S. Ma, and X. Ren, "Jointly learning explainable rules for recommendation with knowledge graph," *The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019*, pp. 1210–1221, 2019.

[70] Q. Ai, V. Azizi, X. Chen, and Y. Zhang, "Learning heterogeneous knowledge base embeddings for explainable recommendation," *Algorithms*, vol. 11, no. 9, p. 137, 2018.

[71] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, "Pathsim: Meta path-based top-k similarity search in heterogeneous information networks," *Proceedings of the VLDB Endowment*, vol. 4, p. 992–1003, aug 2011.

[72] M. G. Ozsoy, D. O'Reilly-Morgan, P. Symeonidis, E. Z. Tragos, N. Hurley, B. Smyth, and A. Lawlor, "MP4Rec: Explainable and accurate top-n recommendations in heterogeneous information networks," *IEEE Access*, 2020.

[73] H. Chen, Y. Li, X. Sun, G. Xu, and H. Yin, "Temporal meta-path guided explainable recommendation," *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, 2021.

[74] R. Sun, X. Cao, Y. Zhao, J. Wan, K. Zhou, F. Zhang, Z. Wang, and K. Zheng, "Multi-modal knowledge graphs for recommender systems," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, Association for Computing Machinery, 2020.

[75] J. Mao, Y. Yao, S. Heinrich, T. Hinz, C. Weber, S. Wermter, Z. Liu, and M. Sun, "Bootstrapping knowledge graphs from images and text," *Frontiers in Neurorobotics*, vol. 13, p. 93, 2019.

[76] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, "Grouplens: An open architecture for collaborative filtering of netnews," in *Proceedings of the 1994 ACM Conference on Computer Supported Cooperative Work*, CSCW '94, (New York, NY, USA), p. 175–186, Association for Computing Machinery, 1994.

[77] M. Balabanović and Y. Shoham, "Fab: Content-based, collaborative recommendation," *Commun. ACM*, vol. 40, p. 66–72, mar 1997.

[78] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, no. 8, pp. 30–37, 2009.

[79] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "BPR: Bayesian personalized ranking from implicit feedback," in *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*, 2009.

[80] Q. Liu, S. Wu, and L. Wang, "Deepstyle: Learning user preferences for visual recommendation," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, (New York, NY, USA), p. 841–844, Association for Computing Machinery, 2017.

[81] S. Wang, Y. Wang, J. Tang, K. Shu, S. Ranganath, and H. Liu, "What your images reveal: Exploiting visual contents for point-of-interest recommendation," in *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, (Republic and Canton of Geneva, CHE), p. 391–400, International World Wide Web Conferences Steering Committee, 2017.

[82] H. T. Nguyen, M. Wistuba, J. Grabocka, L. R. Drumond, and L. Schmidt-Thieme, "Personalized deep learning for tag recommendation," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 10234 LNAI, pp. 186–197, 2017.

[83] P. Gaspar, "User preferences analysis using visual stimuli," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, (New York, NY, USA), p. 436–440, Association for Computing Machinery, 2017.

[84] Y. Zhang, Q. Ai, X. Chen, and W. B. Croft, "Joint representation learning for top-n recommendation with heterogeneous information sources," in *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, CIKM '17, (New York, NY, USA), p. 1449–1458, Association for Computing Machinery, 2017.

[85] Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: Scalable representation learning for heterogeneous networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 135–144, 2017.

[86] V. Bellini, A. Schiavone, T. Di Noia, A. Ragone, and E. Di Sciascio, "Knowledge-aware autoencoders for explainable recommender systems," in *Proceedings of the 3rd Workshop on Deep Learning for Recommender Systems*, (New York, NY, USA), p. 24–31, Association for Computing Machinery, 2018.

[87] F. Yang, N. Liu, S. Wang, and X. Hu, "Towards Interpretation of Recommender Systems with Sorted Explanation Paths," *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2018.

[88] T. Yu, Y. Shen, and H. Jin, "A visual dialog augmented interactive recommender system," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, KDD '19, (New York, NY, USA), p. 157–165, Association for Computing Machinery, 2019.

[89] P. Liu, L. Zhang, and J. A. Gulla, "Dynamic attention-based explainable recommendation with textual and visual fusion," *Information Processing & Management*, p. 102099, 2019.

[90] M. Alshammari, O. Nasraoui, and S. Sanders, "Mining semantic knowledge graphs to add explainability to black box recommender systems," *IEEE Access*, vol. 7, pp. 110563–110579, 2019.

[91] X. Huang, Q. Fang, S. Qian, J. Sang, Y. Li, and C. Xu, "Explainable interaction-driven user modeling over knowledge graph for sequential recommendation," in *Proceedings of the 27th ACM International Conference on Multimedia*, (New York, NY, USA), Association for Computing Machinery, 2019.

[92] X. Du, H. Yin, L. Chen, Y. Wang, Y. Yang, and X. Zhou, "Personalized video recommendation using rich contents from videos," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, pp. 492–505, 2020.

[93] A. Ghazimatin, O. Balalau, R. S. Roy, and G. Weikum, "Prince: Provider-side interpretability with counterfactual explanations in recommender systems," *WSDM 2020 - Proceedings of the 13th International Conference on Web Search and Data Mining*, pp. 196–204, 2020.

[94] Z. Fu, Y. Xian, R. Gao, J. Zhao, Q. Huang, Y. Ge, S. Xu, S. Geng, C. Shah, Y. Zhang, and G. de Melo, *Fairness-Aware Explainable Recommendation over Knowledge Graphs*, p. 69–78. New York, NY, USA: Association for Computing Machinery, 2020.

[95] Y. Xian, Z. Fu, H. Zhao, Y. Ge, X. Chen, Q. Huang, S. Geng, Z. Qin, G. de Melo, S. Muthukrishnan, and Y. Zhang, "Cafe: Coarse-to-fine neural symbolic reasoning for explainable recommendation," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, (New York, NY, USA), p. 1645–1654, Association for Computing Machinery, 2020.

[96] X. Wang, Y. Wang, and Y. Ling, "Attention-Guide Walk Model in Heterogeneous Information Network for Multi-Style Recommendation Explanation," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 6275–6282, 2020.

[97] Z. Li, W. Cheng, H. Xiao, W. Yu, H. Chen, and W. Wang, "You are what and where you are: Graph enhanced attention network for explainable poi recommendation," in

Proceedings of the 30th ACM International Conference on Information & Knowledge Management, Association for Computing Machinery, 2021.

[98] X. Wang, T. Huang, D. Wang, Y. Yuan, Z. Liu, X. He, and T.-S. Chua, "Learning intents behind interactions with knowledge graph for recommendation," in *Proceedings of the Web Conference 2021*, (New York, NY, USA), Association for Computing Machinery, 2021.

[99] Y. Zhu, Y. Xian, Z. Fu, G. de Melo, and Y. Zhang, "Faithfully explainable recommendation via neural logic reasoning," in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2021.

[100] C.-Y. Tai, L.-Y. Huang, C.-K. Huang, and L.-W. Ku, "User-centric path reasoning towards explainable recommendation," in *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, (New York, NY, USA), Association for Computing Machinery, 2021.

[101] S. Tao, R. Qiu, Y. Ping, and H. Ma, "Multi-modal Knowledge-aware Reinforcement Learning Network for Explainable Recommendation," *Knowledge-Based Systems*, vol. 227, p. 107217, 2021.

[102] R. Shimizu, M. Matsutani, and M. Goto, "An explainable recommendation framework based on an improved knowledge graph attention network with massive volumes of side information," *Knowledge-Based Systems*, vol. 239, p. 107970, 2022.

[103] S. Geng, Z. Fu, J. Tan, Y. Ge, G. de Melo, and Y. Zhang, "Path language modeling over knowledge graphsfor explainable recommendation," in *Proceedings of the ACM Web Conference 2022*, WWW '22, (New York, NY, USA), p. 946–955, Association for Computing Machinery, 2022.

[104] Y. Zhao, X. Wang, J. Chen, Y. Wang, W. Tang, X. He, and H. Xie, "Time-aware path reasoning on knowledge graph for recommendation," *ACM Transactions on Information Systems*, vol. 41, no. 2, 2022.

[105] Y. Koren, "Factorization meets the neighborhood: A multifaceted collaborative filtering model," in *Proceedings of the 14th ACM SIGKDD International Conference on*

Knowledge Discovery and Data Mining, KDD '08, (New York, NY, USA), p. 426–434, Association for Computing Machinery, 2008.

[106] D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Advances in Neural Information Processing Systems* (T. Leen, T. Dietterich, and V. Tresp, eds.), vol. 13, MIT Press, 2000.

[107] G. Takács and D. Tikk, "Alternating least squares for personalized ranking," in *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, (New York, NY, USA), p. 83–90, Association for Computing Machinery, 2012.

[108] C. Musto, "Enhanced vector space models for content-based recommender systems," in *Proceedings of the Fourth ACM Conference on Recommender Systems*, RecSys '10, (New York, NY, USA), p. 361–364, Association for Computing Machinery, 2010.

[109] W. H. Lopez Pinaya, S. Vieira, R. Garcia-Dias, and A. Mechelli, "Chapter 11 - autoencoders," in *Machine Learning* (A. Mechelli and S. Vieira, eds.), pp. 193–208, Academic Press, 2020.

[110] L. R. Medsker and L. Jain, "Recurrent neural networks," *Design and Applications*, vol. 5, no. 64-67, p. 2, 2001.

[111] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26–38, 2017.

[112] S. Sedhain, A. K. Menon, S. Sanner, and L. Xie, "Autorec: Autoencoders meet collaborative filtering," in *Proceedings of the 24th international conference on World Wide Web*, pp. 111–112, 2015.

[113] X. Li and J. She, "Collaborative variational autoencoder for recommender systems," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 305–314, 2017.

[114] M. Quadrana, A. Karatzoglou, B. Hidasi, and P. Cremonesi, "Personalizing session-based recommendations with hierarchical recurrent neural networks," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, (New York, NY, USA), p. 130–137, Association for Computing Machinery, 2017.

[115] B. Twardowski, "Modelling contextual information in session-aware recommender systems with neural networks," in *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, (New York, NY, USA), p. 273–276, Association for Computing Machinery, 2016.

[116] C.-Y. Wu, A. Ahmed, A. Beutel, and A. Smola, "Joint training of ratings and reviews with recurrent recommender networks," in *International Conference on Learning Representations*, 2016.

[117] P. Li, Z. Wang, Z. Ren, L. Bing, and W. Lam, "Neural rating regression with abstractive tips generation for recommendation," in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 345–354, 2017.

[118] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proceedings of the tenth ACM international conference on web search and data mining*, pp. 425–434, 2017.

[119] D. Kim, C. Park, J. Oh, S. Lee, and H. Yu, "Convolutional matrix factorization for document context-aware recommendation," in *Proceedings of the 10th ACM conference on recommender systems*, pp. 233–240, 2016.

[120] X. Du, X. He, F. Yuan, J. Tang, Z. Qin, and T.-S. Chua, "Modeling embedding dimension correlations via convolutional neural collaborative filtering," *ACM Transactions on Information Systems*, vol. 37, sep 2019.

[121] J. Tang and K. Wang, "Personalized top-n sequential recommendation via convolutional sequence embedding," in *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, WSDM '18, (New York, NY, USA), p. 565–573, Association for Computing Machinery, 2018.

[122] H. Liang, "Drprofiling: Deep reinforcement user profiling for recommendations in heterogenous information networks," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2020.

[123] X. Wang, Y. Chen, J. Yang, L. Wu, Z. Wu, and X. Xie, "A reinforcement learning framework for explainable recommendation," in *2018 IEEE International Conference on Data Mining (ICDM)*, pp. 587–596, 2018.

[124] X. Chen, L. Yao, J. McAuley, G. Zhou, and X. Wang, “A survey of deep reinforcement learning in recommender systems: A systematic review and future directions,” *arXiv preprint arXiv:2109.03540*, 2021.

[125] H. Cai, V. W. Zheng, and K. C.-C. Chang, “A comprehensive survey of graph embedding: Problems, techniques, and applications,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 9, pp. 1616–1637, 2018.

[126] A. Bordes, N. Usunier, A. Garcia-Durán, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, (Red Hook, NY, USA), p. 2787–2795, 2013.

[127] Y. Lin, Z. Liu, M. Sun, Y. Liu, and X. Zhu, “Learning entity and relation embeddings for knowledge graph completion,” in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI Press, 2015.

[128] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, “Graph attention networks,” *International Conference on Learning Representations*, 2018.

[129] N. Liu, Y. Ge, L. Li, X. Hu, R. Chen, and S.-H. Choi, “Explainable recommender systems via resolving learning representations,” in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, Association for Computing Machinery, 2020.

[130] L. Lovász, “Random walks on graphs,” *Combinatorics, Paul erdos is eighty*, vol. 2, no. 1-46, p. 4, 1993.

[131] A. M. Dai and Q. V. Le, “Semi-supervised sequence learning,” in *Advances in Neural Information Processing Systems* (C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, eds.), vol. 28, Curran Associates, Inc., 2015.

[132] T. Mikolov, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representations in vector space,” *arXiv preprint arXiv:1301.3781*, 2013.

[133] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu, “PathSim: Meta path-based top-k similarity search in heterogeneous information networks,” *Proc. VLDB Endow.*, vol. 4, pp. 992–1003, 2011.

[134] Z. Sun, J. Yang, J. Zhang, A. Bozzon, L.-K. Huang, and C. Xu, “Recurrent knowledge graph embedding for effective recommendation,” in *Proceedings of the 12th ACM Conference on Recommender Systems*, (New York, NY, USA), Association for Computing Machinery, 2018.

[135] T. Ma, L. Huang, Q. Lu, and S. Hu, “Kr-gcn: Knowledge-aware reasoning with graph convolution network for explainable recommendation,” *ACM Transactions on Information Systems*, jan 2022.

[136] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” in *Proceedings of the 22nd ACM International Conference on Multimedia*, p. 675–678, 2014.

[137] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” in *International Conference on Learning Representations*, 2015.

[138] A. Mnih and R. R. Salakhutdinov, “Probabilistic matrix factorization,” *Advances in neural information processing systems*, vol. 20, 2007.

[139] M. Tkalcic and J. Tasic, “Colour spaces: perceptual, historical and applicational background,” in *The IEEE Region 8 EUROCON 2003. Computer as a Tool.*, vol. 1, pp. 304–308 vol.1, 2003.

[140] B. Ferwerda, M. Schedl, and M. Tkalcic, “Predicting personality traits with instagram pictures,” in *Proceedings of the 3rd Workshop on Emotions and Personality in Personalized Systems 2015*, EMPIRE ’15, (New York, NY, USA), p. 7–10, Association for Computing Machinery, 2015.

[141] P. Valdez and A. Mehrabian, “Effects of color on emotions.,” *Journal of experimental psychology: General*, vol. 123, no. 4, p. 394, 1994.

[142] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, “Rethinking the inception architecture for computer vision,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2818–2826, 2016.

[143] S. E. Robertson and K. S. Jones, “Relevance weighting of search terms,” *Journal of the American Society for Information Science*, vol. 27, no. 3, pp. 129–146, 1976.

[144] Q. Le and T. Mikolov, “Distributed representations of sentences and documents,” in *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, ICML’14, p. II–1188–II–1196, JMLR.org, 2014.

[145] K. van de Sande, T. Gevers, and C. Snoek, “Evaluating color descriptors for object and scene recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1582–1596, 2010.

[146] M.-y. Chen and A. Hauptmann, “Mosift: Recognizing human actions in surveillance videos,” *Computer Science Department*, p. 929, 2009.

[147] K. He, X. Zhang, S. Ren, and J. Sun, “Spatial pyramid pooling in deep convolutional networks for visual recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 9, pp. 1904–1916, 2015.

[148] F. Zhang, N. J. Yuan, D. Lian, X. Xie, and W.-Y. Ma, “Collaborative knowledge base embedding for recommender systems,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD ’16, (New York, NY, USA), p. 353–362, Association for Computing Machinery, 2016.

[149] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255, 2009.

[150] Y. Kim, “Convolutional neural networks for sentence classification,” in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (Doha, Qatar), pp. 1746–1751, Association for Computational Linguistics, Oct. 2014.

[151] B. Kveton, C. Szepesvari, Z. Wen, and A. Ashkan, “Cascading bandits: Learning to rank in the cascade model,” in *International Conference on Machine Learning*, 2015.

[152] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi, “A survey of methods for explaining black box models,” *ACM Computing Surveys*, vol. 51, aug 2018.

[153] “Movielens.” https://grouplens.org/datasets/movielens/. Accessed: 2024-01-12.

[154] “Freebase.” https://developers.google.com/freebase/. Accessed: 2024-01-12.

[155] J. Huang, W. X. Zhao, H. Dou, J.-R. Wen, and E. Y. Chang, "Improving sequential recommendation with knowledge-enhanced memory networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, (New York, NY, USA), p. 505–514, Association for Computing Machinery, 2018.

[156] W. X. Zhao, G. He, H. Dou, J. Huang, S. Ouyang, and J.-R. Wen, "Kb4rec: A dataset for linking knowledge bases with recommender systems," 2018.

[157] "Microsoft satori." http://searchengineland.com/library/bing/bing-satori. Accessed: 2021-07-01.

[158] "Amazon product data." http://jmcauley.ucsd.edu/data/amazon. Accessed: 2023-12-23.

[159] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: A collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD '08, (New York, NY, USA), p. 1247–1250, Association for Computing Machinery, 2008.

[160] "Hetrec 2011." https://grouplens.org/datasets/hetrec-2011/. Accessed: 2023-12-23.

[161] "Yelp open dataset." https://www.yelp.com/dataset. Accessed: 2023-12-23.

[162] "Book-crossing dataset." https://www.kaggle.com/datasets/somnambwl/bookcrossing-dataset. Accessed: 2023-12-23.

[163] "Wsdm - kkbox's music recommendation challenge." https://www.kaggle.com/c/kkbox-music-recommendation-challenge/data. Accessed: 2023-12-23.

[164] "Cn-dbpedia." http://kw.fudan.edu.cn/cndbpedia/search/. Accessed: 2023-12-23.

[165] B. Xu, Y. Xu, J. Liang, C. Xie, B. Liang, W. Cui, and Y. Xiao, "Cn-dbpedia: A never-ending chinese knowledge extraction system," in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems*, 2017.

[166] B. Abdollahi and O. Nasraoui, "Explainable matrix factorization for collaborative filtering," in *Proceedings of the 25th International Conference Companion on World Wide Web*, (Republic and Canton of Geneva, CHE), p. 5–6, International World Wide Web Conferences Steering Committee, 2016.

[167] B. Abdollahi and O. Nasraoui, "Using explainability for constrained matrix factorization," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, (New York, NY, USA), p. 79–83, Association for Computing Machinery, 2017.

[168] R. C. Fong and A. Vedaldi, "Interpretable explanations of black boxes by meaningful perturbation," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3449–3457, 2017.

[169] L. van der Maaten and G. E. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008.

[170] S. Serrano and N. A. Smith, "Is attention interpretable?," in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, (Florence, Italy), pp. 2931–2951, Association for Computational Linguistics, July 2019.

[171] S. Subramanian, B. Bogin, N. Gupta, T. Wolfson, S. Singh, J. Berant, and M. Gardner, "Obtaining faithful interpretations from compositional neural networks," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, (Online), pp. 5594–5608, Association for Computational Linguistics, July 2020.

[172] G. Navarro, "A guided tour to approximate string matching," *ACM Computing Surveys*, vol. 33, p. 31–88, mar 2001.

[173] T. Markchom and H. Liang, "Augmenting visual information in knowledge graphs for recommendations," in *26th International Conference on Intelligent User Interfaces*, p. 475–479, 2021.

[174] H. Wang, F. Zhang, X. Xie, and M. Guo, "DKN: Deep knowledge-aware network for news recommendation," in *Proceedings of the 2018 World Wide Web Conference*, WWW '18, (Republic and Canton of Geneva, CHE), p. 1835–1844, International World Wide Web Conferences Steering Committee, 2018.

[175] B. Hu, C. Shi, W. X. Zhao, and P. S. Yu, "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*, p. 1531–1540, 2018.

[176] X. Wang, D. Bo, C. Shi, S. Fan, Y. Ye, and S. Y. Philip, "A survey on heterogeneous graph embedding: methods, techniques, applications and sources," *IEEE Transactions on Big Data*, vol. 9, no. 2, pp. 415–436, 2022.

[177] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems* (C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, eds.), vol. 26, Curran Associates, Inc., 2013.

[178] R. C. González and R. E. Woods, *Digital image processing, 3rd Edition*. Pearson Education, 2008.

[179] S. Gkelios, A. Sophokleous, S. Plakias, Y. Boutalis, and S. A. Chatzichristofis, "Deep convolutional features for image retrieval," *Expert Systems with Applications*, vol. 177, p. 114940, 2021.

[180] M. S. Hasan *et al.*, "An application of pre-trained cnn for image classification," in *2017 20th international conference of computer and information technology (ICCIT)*, pp. 1–6, IEEE, 2017.

[181] S. Mascarenhas and M. Agarwal, "A comparison between vgg16, vgg19 and resnet50 architecture frameworks for image classification," in *2021 International conference on disruptive technologies for multi-disciplinary research and applications (CENTCON)*, vol. 1, pp. 96–99, IEEE, 2021.

[182] G.-H. Liu and J.-Y. Yang, "Content-based image retrieval using color difference histogram," *Pattern recognition*, vol. 46, no. 1, pp. 188–198, 2013.

[183] P. Chhabra, N. K. Garg, and M. Kumar, "Content-based image retrieval system using orb and sift features," *Neural Computing and Applications*, vol. 32, pp. 2725–2733, 2020.

[184] P. Loncomilla, J. Ruiz-del Solar, and L. Martínez, "Object recognition using local invariant features for robotic applications: A survey," *Pattern Recognition*, vol. 60, pp. 499–514, 2016.

[185] M. Schwarz, H. Schulz, and S. Behnke, "Rgb-d object recognition and pose estimation based on pre-trained convolutional neural network features," in *2015 IEEE international conference on robotics and automation (ICRA)*, pp. 1329–1335, IEEE, 2015.

[186] E. Rosten and T. Drummond, “Machine learning for high-speed corner detection,” in *Computer Vision – ECCV 2006* (A. Leonardis, H. Bischof, and A. Pinz, eds.), (Berlin, Heidelberg), pp. 430–443, Springer Berlin Heidelberg, 2006.

[187] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, “Brief: Binary robust independent elementary features,” in *Computer Vision–ECCV 2010: 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV 11*, pp. 778–792, Springer, 2010.

[188] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “Imagenet classification with deep convolutional neural networks,” *Commun. ACM*, vol. 60, p. 84–90, may 2017.

[189] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[190] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, “Grad-cam: Visual explanations from deep networks via gradient-based localization,” in *Proceedings of the IEEE international conference on computer vision*, pp. 618–626, 2017.

[191] M. Tan and Q. Le, “Efficientnet: Rethinking model scaling for convolutional neural networks,” in *International conference on machine learning*, pp. 6105–6114, PMLR, 2019.

[192] C. Celik and H. S. Bilge, “Content based image retrieval with sparse representations and local feature descriptors : A comparative study,” *Pattern Recognition*, vol. 68, pp. 1–13, 2017.

[193] H. Liang and T. Baldwin, “A probabilistic rating auto-encoder for personalized recommender systems,” in *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*, pp. 1863–1866, 2015.

[194] I. Cantador, P. Brusilovsky, and T. Kuflik, “2nd workshop on information heterogeneity and fusion in recommender systems (HetRec 2011),” in *Proceedings of the 5th ACM conference on Recommender systems*, 2011.

[195] “Grouplens: Social computing research at the university of minnesota.” https://grouplens.org. Accessed: 2023-12-23.

[196] "Rotten tomatoes." http://www.rottentomatoes.com. Accessed: 2023-12-23.

[197] "Imdb." http://www.imdb.com. Accessed: 2023-12-23.

[198] "Omdb api: The open movie database." http://www.omdbapi.com. Accessed: 2023-12-23.

[199] R. He and J. McAuley, "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*, p. 507–517, 2016.

[200] "Opencv." Accessed: 2023-12-07.

[201] "Caffe." `https://github.com/BVLC/caffe`. Accessed: 2023-12-07.

[202] T. Markchom, H. Liang, and J. Ferryman, "Scalable and explainable visually-aware recommender systems," *Knowledge-Based Systems*, vol. 263, p. 110258, 2023.

[203] B. Abdollahi and O. Nasraoui, "Using explainability for constrained matrix factorization," in *Proceedings of the Eleventh ACM Conference on Recommender Systems*, p. 79–83, 2017.

[204] G. Peake and J. Wang, "Explanation mining: Post hoc interpretability of latent factor models for recommendation systems," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, p. 2060–2069, 2018.

[205] Y. Yang, Z. Guan, J. Li, W. Zhao, J. Cui, and Q. Wang, "Interpretable and efficient heterogeneous graph convolutional network," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2021.

[206] M. Zhang, G. Wang, L. Ren, J. Li, K. Deng, and B. Zhang, "Metonr: A meta explanation triplet oriented news recommendation model," *Knowledge-Based Systems*, vol. 238, feb 2022.

[207] V. Jagadeesh, R. Piramuthu, A. Bhardwaj, W. Di, and N. Sundaresan, "Large scale visual recommendations from street fashion images," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '14, (New York, NY, USA), p. 1925–1934, Association for Computing Machinery, 2014.

[208] H. Liang, Z. Liu, and T. Markchom, "Relation-aware blocking for scalable recommendation systems," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, CIKM '22, (New York, NY, USA), p. 4214–4218, Association for Computing Machinery, 2022.

[209] H. Liang and T. Markchom, "TNE: A general time-aware network representation learning framework for temporal applications," *Knowledge-Based Systems*, vol. 240, mar 2022.

[210] T. Markchom, H. Liang, and J. Ferryman, "Explainable meta-path based recommender systems," *ACM Trans. Recomm. Syst.*, sep 2023. Just Accepted.

[211] S. Yun, M. Jeong, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks," in *Advances in Neural Information Processing Systems*, pp. 11960–11970, 2019.

[212] S. Yun, M. Jeong, S. Yoo, S. Lee, S. S. Yi, R. Kim, J. Kang, and H. J. Kim, "Graph transformer networks: Learning meta-path graphs to improve gnns," *Neural Networks*, vol. 153, pp. 104–119, 2022.

[213] Y. Chang, C. Chen, W. Hu, Z. Zheng, X. Zhou, and S. Chen, "Megnn: Meta-path extracted graph neural network for heterogeneous graph representation learning," *Knowledge-Based System*, vol. 235, jan 2022.

[214] D. A. Smith and J. Eisner, "Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies," in *Proceedings of the Workshop on Statistical Machine Translation*, StatMT '06, (USA), p. 23–30, Association for Computational Linguistics, 2006.

[215] Y. Kim, "Sequence-to-sequence learning with latent neural grammars," in *Proceedings of the 35th Conference on Neural Information Processing Systems (NeurIPS 2021)*, 2021.

[216] J. S. Adelman, G. D. Brown, and J. F. Quesada, "Contextual diversity, not word frequency, determines word-naming and lexical decision times," *Psychological Science*, vol. 17, no. 9, pp. 814–823, 2006.

[217] K. S. Tai, R. Socher, and C. D. Manning, "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), (Beijing, China), pp. 1556–1566, Association for Computational Linguistics, July 2015.

[218] J. K. Baker, "Trainable grammars for speech recognition," *The Journal of the Acoustical Society of America*, vol. 65, no. S1, pp. S132–S132, 1979.

[219] S. J. Rennie, E. Marcheret, Y. Mroueh, J. Ross, and V. Goel, "Self-critical sequence training for image captioning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[220] X. Yu, X. Ren, Y. Sun, Q. Gu, B. Sturt, U. Khandelwal, B. Norick, and J. Han, "Personalized entity recommendation: A heterogeneous information network approach," *WSDM 2014 - Proceedings of the 7th ACM International Conference on Web Search and Data Mining*, pp. 283–292, 2014.

[221] H. Zhao, Q. Yao, J. Li, Y. Song, and D. L. Lee, "Meta-graph based recommendation fusion over heterogeneous information networks," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, (New York, NY, USA), p. 635–644, Association for Computing Machinery, 2017.

[222] B. Hu, C. Shi, W. X. Zhao, and T. Yang, "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*, CIKM '18, (New York, NY, USA), p. 1683–1686, Association for Computing Machinery, 2018.

[223] Z. Wang, H. Liu, Y. Du, Z. Wu, and X. Zhang, "Unified embedding model over heterogeneous information network for personalized recommendation," in *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, IJCAI'19, p. 3813–3819, AAAI Press, 2019.

[224] Z. Han, F. Xu, J. Shi, Y. Shang, H. Ma, P. Hui, and Y. Li, "Genetic meta-structure search for recommendation on heterogeneous information network," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, (New York, NY, USA), p. 455–464, Association for Computing Machinery, 2020.

[225] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, p. 1243–1252, JMLR.org, 2017.

[226] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.

[227] P. Ramachandran, P. Liu, and Q. Le, "Unsupervised pretraining for sequence to sequence learning," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, (Copenhagen, Denmark), pp. 383–391, Association for Computational Linguistics, Sept. 2017.

[228] W. Wang, W. Jiao, Y. Hao, X. Wang, S. Shi, Z. Tu, and M. Lyu, "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)*, (Dublin, Ireland), pp. 2591–2600, Association for Computational Linguistics, May 2022.

[229] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "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)*, (Doha, Qatar), pp. 1724–1734, Association for Computational Linguistics, Oct. 2014.

[230] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, (Cambridge, MA, USA), p. 3104–3112, MIT Press, 2014.