NEAR: Non-Supervised Explainability Architecture for Accurate Review-Based Collaborative Filtering

Reinald Adrian Pugoy, *Student Member, IEEE* and Hung-Yu Kao, *Member, IEEE*

Abstract—There is a critical issue in explainable recommender systems that compounds the challenges of explainability yet is rarely tackled: the lack of ground-truth explanation texts for training. It is unrealistic to expect every user-item pair in a dataset to have a corresponding target explanation. Hence, we pioneer the first non-supervised explainability architecture for review-based collaborative filtering (called NEAR) as our novel contribution to the theory of explanation construction in recommender systems. While maintaining excellent recommendation performance, our approach reformulates explainability as a non-supervised (i.e., unsupervised and self-supervised) explanation generation task. We formally define two explanation types, both of which NEAR can produce. An invariant explanation, fixed for all users, is based on the unsupervised extractive summary of an item's reviews via embedding clustering. Meanwhile, a variant explanation, personalized for a specific user, is a sentence-level text generated by our customized Transformer conditioned on every user-item-rating tuple and artificial ground-truth (self-supervised label) from one of the invariant explanation's sentences. Our empirical evaluation illustrates that NEAR's rating prediction accuracy is better than the other state-of-the-art baselines. Moreover, experiments and assessments show that NEAR-generated variant explanations are more personalized and distinct than those from other Transformer-based models, and our invariant explanations are preferred over those from other contemporary models in real life.

Index Terms—Collaborative Filtering, Explainability, Rating Prediction, Recommender Systems, Summarization, Transformer

1 Introduction

Collaborative filtering (CF) approaches are the most dominant and outstanding models in recommender systems. In essence, CF aims to learn accurate representations of users and items, respectively denoting user preferences and item characteristics [1], [2]. The earliest CF models learned these representations based on user-given numeric ratings. However, employing them is a clear case of oversimplifying user and item properties [3], [4]. In this regard, review texts have been utilized to alleviate this issue. The main advantage of employing reviews as the feature source is that they can cover the multi-faceted substance of user opinions. Since users can freely discuss their reasons for the ratings that they give to items, reviews contain a large quantity of rich latent information that cannot be otherwise acquired exclusively from ratings [1].

Nonetheless, a common limitation exists for most neural review-based CF; despite widespread usage, they have largely remained black-boxes [5]. The inherent black-box nature of neural networks (NN) opaques the explainability behind predictions such that their complex architecture have obscured the underlying decision-making process [6], [7], [8]. Providing explanations is said to be necessary as these can improve user satisfaction, confidence, and trust in a recommender system, which can aid potential clients to make better, faster, and informed decisions [7], [9], [10].

Considering the aforementioned, explainable recommendation has increasingly attracted the attention of the academic, industrial, and research communities [11]. It is primarily founded on the premise of designing new recommendation algorithms that can provide explanations [12]. Yet, we argue that a critical issue that compounds the challenges of explainability is rarely tackled: the lack of ground-truth explanation texts required for training. A typical dataset consists of a user ID, item ID, rating, and review. But, it is unrealistic to expect that every user-item pair in such large datasets has a corresponding target explanation. Obtaining these targets from scratch is cumbersome, labor-intensive, and time-consuming. As an unfortunate consequence, relying only on explanation-labeled datasets severely limits the domains that a recommender model can be trained on.

Technically speaking, word-level and review-level explanations appear to address the issue at hand recently. They are dependent on the application of vanilla attention to words and reviews [1], [13], [14], [15], [16]. High-scoring words and reviews are then selected to serve as explanations. Still, we contend that both explanation types may not resemble real-life explanations and are found less acceptable by humans [17]. As illustrated in Table 1, a review-level explanation is exactly the same as the review with the highest attention weight, inadvertently disregarding other useful sentences from lower-weighted reviews. On the contrary, a word-level explanation (underlined in Table 1) may appear fragmented and not be intelligible enough in an

- *Both authors are with the Intelligent Knowledge Management Lab, Department of Computer Science and Information Engineering, National Cheng Kung University, Tainan City, Taiwan.*
  E-mail: p78077040@ncku.edu.tw, hykao@mail.ncku.edu.tw
- *R.A. Pugoy is also with the Faculty of Information and Communication Studies, University of the Philippines Open University, Los Baños, Philippines.*
  E-mail: rdpugoy@up.edu.ph

TABLE 1
Illustration of different explanation types.

	Sample Item Reviews
(1)	*I've tried several of the various mouse control methods. You end up with a live mouse stuck to a piece of plastic by a thick gummy glue. The mouse squirms and may even try to gnaw its own leg off to get free. Not humane and not a pretty site. This is the absolute worst method.*
(2)	*You bait these traps by unscrewing a very small plug and putting some bait on it, I used peanut butter, and screwing it back in. You move a lever to set the trap and it can only be set once so when I accidentally dropped one trap it triggered and it was no longer usable.*
(3)	*The only difference between this trap and a regular spring-loaded trap is that this one has a cover over it so you don't have to see the dead mouse or look at it. That is the only advantage, yet you're paying several dollars per trap more. However, I think they are incredibly inhumane and an awful way to kill mice and rats.*
	Sample Explanations from Other Models
Review-Level	*I've tried several of the various mouse control methods. You end up with a live mouse stuck to a piece of plastic by a thick gummy glue. The mouse squirms and may even try to gnaw its own leg off to get free. Not humane and not a pretty site. This is the absolute worst method.*
Word-Level	*I've tried several of the various mouse control methods. You end up with a live mouse stuck to a piece of plastic by a thick gummy glue. The mouse squirms and may even try to gnaw its own leg off to get free. Not humane and not a pretty site. This is the absolute worst method.*
Sentence-Level (NLG)	*Not humane and not a pretty site.*
	Sample Explanations from Our Proposed Model
Invariant	*You bait these traps by unscrewing a very small plug and putting some bait on it, I used peanut butter, and screwing it back in. The only difference between this trap and a regular spring-loaded trap is that this one has a cover over it so you don't have to see the dead mouse or look at it. That is the only advantage, yet you're paying several dollars per trap more. The mouse squirms and may even try to gnaw its own leg off to get free. Not humane and not a pretty site.*
Variant (Pred. Rating of 5)	*You bait these traps by unscrewing a very small plug and putting some bait on it, I used peanut butter, and screwing it back in.*
Variant (Pred. Rating of 3)	*That is the only advantage, yet you're paying several dollars per trap more.*
Variant (Pred. Rating of 1)	*They are not humane and not a pretty site.*

actual recommendation scenario. Another consequent issue that arises is the lack of user personalization, i.e., the same explanation will be provided to the user regardless of his preference toward the item.

Personalized explanations are still relatively uncommon yet have tremendous potential in explainable recommender systems. Based on the principles of natural language generation (NLG), they can be implemented as RNNs [18] and lately, the Transformer [12]. However, both implementations are also heavily reliant on manually-crafted ground-truth explanations. The RNN-based model requires them for conditioning the sequence decoder, and the Transformer-based approach has to encode them as part of the input sequence.

Therefore, in this paper, we propose an original CF framework called **NEAR** (or Non-Supervised Explainability Architecture for Accurate Review-Based Collaborative Filtering). While maintaining excellent rating prediction accuracy, our approach reformulates explainability as a non-supervised (i.e., unsupervised and self-supervised) generation of explanation texts, providing a unique pipeline for both recommendation and explanation. In the context of recommender systems, we formally theorize and define two types of explanations for the first time, both of which NEAR can generate. Our paper is offering a novel contribution to the theory of explanation construction in recommender systems.

Invariant explanations are summary-level texts that are fixed or the same for all users. Our implementation is based on the unsupervised extractive summarization of an item's received reviews. Essentially, the sentences from the item's review set are encoded using a pre-trained language model and later fed to the embedding clustering algorithm. Unlike a review-level or word-level explanation, a summary-level explanation is composed of informative statements gathered from different reviews. As a beneficial side-effect, this also deals with the user cold-start problem.

Variant explanations are sentence-level texts that are personalized and conditioned on every user-item-rating tuple. Specifically, such explanations are geared toward the user's desire for control, which is key in providing a personalized experience [19]. This desire lies on the premise that when a user gets something tailored to him, this makes him feel to be more in control, serving as a fundamental motivator of a person's decisions [20]. Hence, we treat the predicted rating as the approximate indicator of user interest toward an item; it serves as the rough basis of a personalized explanation that reflects the rating-sentiment (i.e., rating-justifying explanation). Our greedy approach involves selecting artificial ground-truth explanations (self-supervised labels) for every user-item pair, thereby rendering manual crafting unnecessary. In effect, this leads to speedup in ground-truth selection and decision-making process.

Furthermore, these are the other major contributions of our paper:

- To the best of our knowledge, we are the first to stress the importance of non-supervised approaches for producing recommendation explanations.
- Summarization for explainable recommendation is a largely unexplored area. We pioneer the first CF model

that produces explanations based on unsupervised summarization.

- Likewise, the Transformer is still relatively under-utilized for personalized explanations. To the extent of our knowledge also, our model is one of the first to enable the Transformer for explainability.
- We propose three types of appropriate evaluation metrics for assessing explanations. First is an automatic metric that measures the relative personalization among explanation-generating methods. Second is another metric that measures how much the explanation justifies the rating it is supposed to reflect, based on the user's desire for control premise. Third is a metric based on human assessments, i.e., a comprehensive set of criteria that measures the real-life acceptability of explanation texts.

2 Related Work

2.1 Foundational Neural Recommendation

The principle behind CF is that leveraging collaborative behaviors of all existing users is a prerequisite for predicting the behavior of a new/target user [21]. Designing a CF model involves two crucial steps: learning user/item representations (representation learning) and modeling user-item interactions based on those representations (interaction modeling) [22]. The general objective of representation learning is to learn user and item embedding matrices P and Q, while for interaction modeling, it is to estimate the potential user preference toward a specific item [21].

One of the fundamental works in the utilization of NN in CF is called neural collaborative filtering (NCF) [23]. NCF, implemented initially for implicit feedback data-driven CF, learns non-linear, flexible, and more abstract interactions between users and items by employing multilayer perceptron (MLP) layers as its interaction function. An MLP-based interaction function overcomes the limitations of an inner product-based interaction function. The latter is said to be sub-optimal to learn rich, complicated patterns from real-world data [23].

DeepCoNN is the first deep learning-based model representing users and items from reviews in a joint manner [24]. The model consists of two parallel networks powered by convolutional neural networks (CNN). One network learns user behavior by examining all reviews he has written, and the other network models item properties by exploring all reviews it has received. A shared layer connects these two networks, and factorization machines capture user-item interactions. Another significant model is NARRE, which shares several similarities with DeepCoNN. NARRE is also composed of two parallel CNN-based networks that are uniquely incorporated with the review-level attention mechanism [1]. The said mechanism distinguishes each review's usefulness or contribution based on attention weights. As a side-effect, this also leads to review-level explanations; reviews with the highest attention scores are presented as explanations. These weights are then incorporated into the representations of users and items to enhance embedding quality.

2.2 Graph-Based Recommendation

Learning user and item representations undoubtedly lies at the center of modern recommendation [25]. In order to further improve the quality of such representations and better capture the CF effect, Wang et al. [25] proposed to encode the collaborative signal (latent in user-item interactions) during the embedding process. Their approach gave birth to NGCF, the first graph-based CF model that exploits the bipartite user-item graph structure by propagating embeddings on it. Employing graph convolutional networks (GCN) injects the collaborative signal to the user and item representations. Other graph-based CF models include DGCF [26] and LightGCN [27]. While DGCF is similar to the models mentioned, it was developed for sequential data and dynamic recommendation. On the other hand, both NGCF and LightGCN are used for traditional, offline recommendation and are driven by implicit feedback data (similar to NCF). LightGCN is arguably a simplified version of the original NGCF model and also propagates on the user-item graph. But, unlike NGCF's non-linear activation to obtain the final embeddings, LightGCN uses the weighted sum of embeddings for easier implementation and training.

2.3 Attention-Based Recommendation

Other relevant studies involve AHN [28], CAML [5], DAML [14], D-Attn [16], DER [11], HUITA [29], and MPCN [2]. These employ various attention mechanisms to distinguish informative parts of a given data sample, resulting in simultaneous accuracy and explainability improvements. D-Attn integrates global and local attention to score each word to determine its relevance in a review text. DAML utilizes CNN's local and mutual attention to learn review features, and HUITA incorporates a hierarchical, three-tier attention network. MPCN is similar to NARRE, but the former does not rely on convolutional layers. Instead, it introduces a review-by-review pointer-based mechanism that is co-attentive in nature to model user-item relationships. CAML, whose design is based on an encoder-selector-decoder architecture, also relies on hierarchical co-attention to effectively learn the cross-knowledge transfer between the recommendation and explanation tasks. Furthermore, AHN proposes a multi-hierarchical paradigm that recognizes item reviews and user reviews by means of co-attention. To further enhance user experience, DER produces explanation texts by integrating gated recurrent units (GRU), sentence-level CNN, and personalized attention mechanisms.

2.4 BERT-Based Recommendation

The previous paragraph enumerates models that usually take advantage of CNNs as automatic review feature extractors that are coupled with mainstream word embeddings to build user and item representations. Notwithstanding, CNN-based approaches fail to consider global context and word frequency information, crucial in affecting recommendation performance [30], [31]. In this regard, NCEM [13] and BENEFICT [15] replaced the CNN with a pre-trained BERT model in their parallel user/item networks. BERT's advantage lies in its full retention of global context and word frequency information [13]. This integration has led these models to achieve rating prediction performance that is on par with or better than the CNN-based recommenders.

For explainability, NCEM similarly adopts NARRE's review-level attention. On the other hand, BENEFICT utilizes BERT's self-attention weights in conjunction with a solution to the maximum subarray problem (MSP). BENEFICT's approach produces an explanation based on a subarray of contiguous tokens with the largest possible sum of self-attention weights. Moreover, we have proposed another BERT-powered model in a prior study. ESCOFILT provides static, ratio-based summary for both representation and explanation [17]. The study's results are encouraging as they show the promise of extractive summarization for explainability.

2.5 Personalized Explanations

A personalized experience is providing something tailored to the user to deal with information overload and his desire for control [19]. At its most basic sense, the requirement for generating personalized explanations, also called attribute-conditioned explanations, involves the encoding of user and item IDs [12]. The earliest approaches followed a modified version of the seq2seq architecture for explanation generation [18], review generation [32], and tip generation [33]. In the said architecture, the sequence encoder is replaced by the attribute encoder, typically implemented as an MLP to encode both user and item IDs into a context vector. An RNN subsequently decodes a word vector from the said context vector, with each word embedding of the ground-truth text fed at every time step during the training process. Particularly, to the best of our knowledge, NRT [33] is the first model to simultaneously predict ratings and generate explanations in the form of tips. The NETE framework [18] also provides both personalized recommendation and explanation by learning sentence templates from data and generates template-controlled sentences about a specific feature whose embedding is also fed to the RNN decoder. Furthermore, Zhao et al. [34] developed a similar model for explainable song recommendation. Instead of encoding user and item IDs, their model encodes a sequence of words serving as the ground-truth (e.g., song name, singer name, music tags, user status/occupation) to generate an explanation for a particular song and user.

In explanation generation, there are two types of inputs that need to be dealt with: IDs and words. Without a proper solution, these clearly heterogeneous inputs will be placed in the same semantic space as the words, and in effect, the IDs will be treated as out-of-vocabulary (OOV) tokens. In light of this, the PETER model primarily deals with these aforementioned issues; it elevates the old architecture (specifically NETE) to a Transformer-based model for personalized, NLG-derived explanation texts [12]. It should be noted that the generation of personalized prediction explanations is still relatively young in recommender systems research, let alone the utilization of the Transformer that is well-known for its strong language modeling ability on a wide variety of applications [35], [36]. Being the first of its kind, PETER is a multi-training pipeline that jointly performs rating prediction, explanation generation, and context prediction tasks on top of the Transformer. Its inputs include the user ID, item ID, and ground-truth explanation text, with the latter being manually extracted first from user reviews. Its authors specifically introduced the context prediction task to map user and item IDs onto words, effectively endowing them with semantic meanings. This particular approach prevents the former from being treated as OOV. Moreover, PETER has made available the provision of encoding multiple features in the input sequence; a feature can be thought of as a single word to guide the model to express specific topics.

2.6 Observations

Based on the related literature above, these are our observations that seem to hold true:

- **For attention-based and BERT-based recommenders:**
  - Usually, their CF design has two parallel networks, each integrated with either CNN or BERT for building user and item representations.
  - Their main advantage is that they can provide excellent rating prediction accuracy. At the same time, the majority of them settle for explanations that may not be acceptable in real life [17].
  - Explanations are mere side-effects of applying the attention mechanism, whose original intention is to produce better embeddings by determining the user/item features' degree of usefulness. Though not the primary goal, their unsupervised nature (i.e., non-reliance on ground-truth texts) is a by-product of this process.
  - Tackling explainability improves prediction and recommendation performance consequentially.
- **For recommenders with personalized explanation generation:**
  - Their key advantage is that they can generate personalized explanations that are contingent on user preferences. However, accurate recommendation is not their priority.
  - Primarily based on a modified seq2seq architecture, they need ground-truth explanations for the training process. Because there are not many public datasets that include these target labels, this severely limits the domains that the model can be trained on.
  - Improving the quality of personalized explanation involves introducing more features to the Transformer's input sequence. However, there are two weaknesses to this strategy. First, the features also have to be manually crafted from the dataset. Second, including more features means increasing the computational size and trainable parameters of the neural network model.

Overall, there appears to be an unintended trade-off between accuracy and explainability. Another trend we notice is that invariant and variant explanations are never differentiated, if at all. We intend to address this research gap by demonstrating that our proposed framework addresses the aforementioned points, thereby equally giving importance to the explainability and accuracy aspects of a review-based CF.

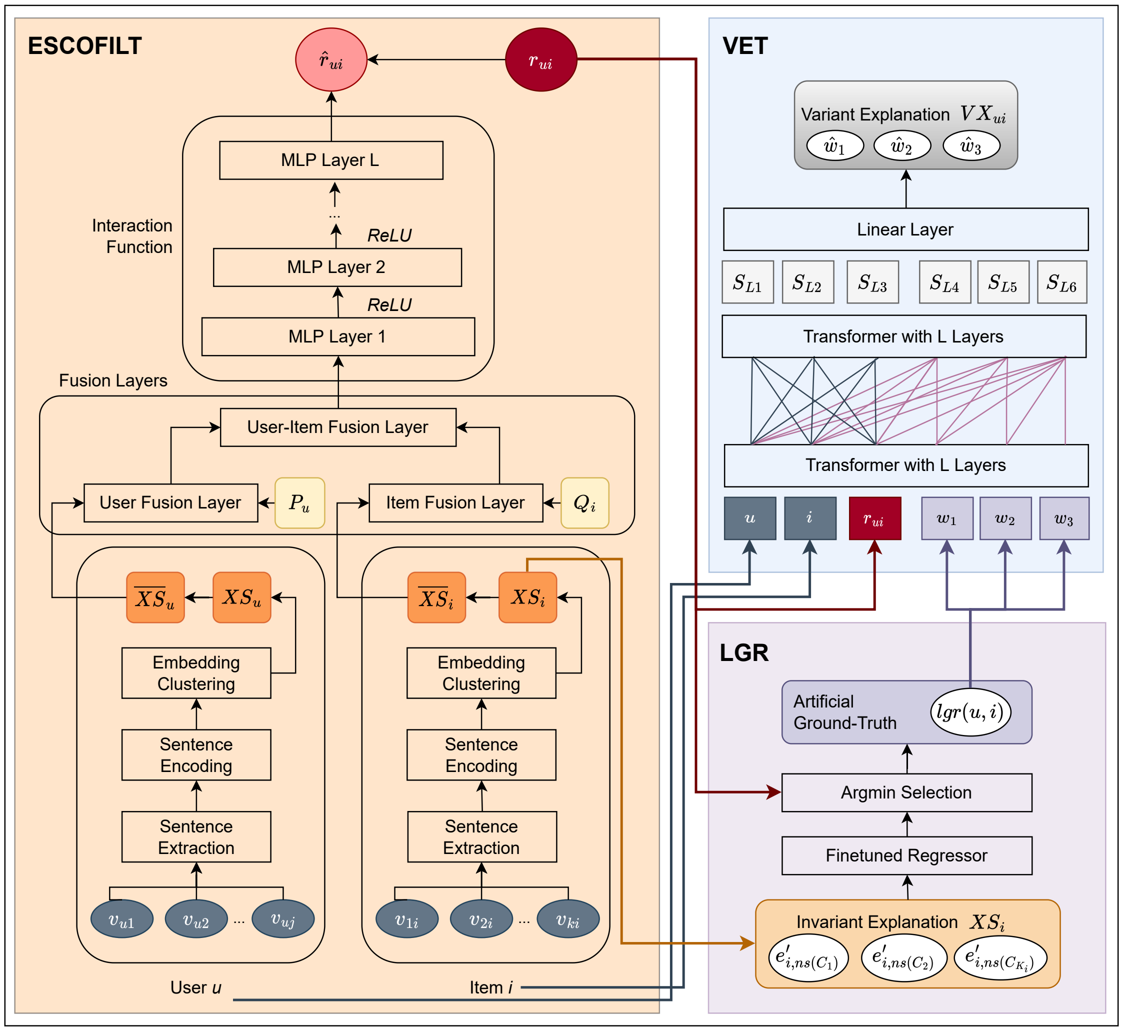

Fig. 1. The proposed NEAR architecture.

3 Methodology

3.1 Problem Formulation and Notation

The training dataset τ consists of N tuples, with the latter denoting the domain-specific dataset size. Each tuple follows this form: (u, i, r_{ui}, v_{ui}) where r_{ui} and v_{ui} respectively refer to the ground-truth rating and review accorded by user u to item i. Let $RV_u = \{v_{u1}, v_{u2}, ..., v_{uj}\}$ be the set of all j reviews written by user u. Likewise, let $RV_i = \{v_{1i}, v_{2i}, ..., v_{ki}\}$ be the set of all k reviews received by item i. Both RV_u and RV_i are obtained from scanning τ itself. NEAR's input is a user-item pair (u, i) from each tuple in τ. We particularly feed RV_u and RV_i to the model as the initial inputs.

Fig. 1 illustrates the architecture of NEAR, which has three distinct and separately trained modules: **Extractive Summarization-Based Collaborative Filtering** (ESCOFILT), **Label-Giving Regressor** (LGR), and **Variant Explanation Transformer** (VET), whose details are all summarized in Table 2. ESCOFILT produces both the predicted rating and the invariant explanation. LGR is a supporting module that decides which specific sentence from the invariant explanation can best serve as the artificial ground-truth explanation. Lastly, VET is our customized Transformer that generates the variant explanation based on the user ID, item ID, rating, and LGR-produced artificial target. NEAR has three outputs. First is the predicted rating $\hat{r}_{ui} \in \mathbb{R}$ that user u may give to item i. In this instance, the rating prediction task is denoted by:

$$predict(u, i) = (RV_u, RV_i) \rightarrow \hat{r}_{ui} \tag{1}$$

The two other outputs are the prediction explanations, i.e., the invariant explanation XS_i and the variant explanation VX_{ui}. The former depicts the item's extractive summary that both represents and explains i, while the latter corresponds to the preference of u toward i. Thus, the explanation task can be described as:

$$explain(\hat{r}_{ui} \mid u, i) = (RV_u, RV_i) \rightarrow (XS_i, VX_{ui}) \tag{2}$$

3.2 ESCOFILT: Extractive Summarization-Based Collaborative Filtering

From this point to Section 3.2.2, we will only discuss the item modeling procedure since it is nearly identical to user modeling, with their inputs as the key difference. ESCOFILT is trained using the Mean Squared Error (MSE) as the

TABLE 2
The different components of NEAR and description of their tasks during training.

Component	Task	Input	Output
ESCOFILT	Rating Prediction, Invariant Explanation Generation	User u, Item i *(respectively represented by the set of reviews RV_u and RV_i)*	Predicted rating $\hat{r}_{ui}$, Invariant explanation XS_i
LGR	Artificial Ground-Truth Selection	Invariant explanation XS_i, Ground-truth rating r_{ui}	Artificial ground-truth $lgr(u,i)$, *(i.e., a sentence in XS_i)*
VET	Variant Explanation Generation	User u, Item i, Artificial ground-truth $lgr(u,i)$, Ground-truth rating r_{ui}	Variant explanation VX_{ui}

objective function:

$$MSE = \frac{1}{|\tau|} \sum_{u,i \in \tau} (r_{ui} - \hat{r}_{ui})^2 \tag{3}$$

3.2.1 Sentence Extraction and Encoding

Initially, the reviews in RV_i are concatenated together to form a single document. A sentence segmentation method called Sentencizer (by spaCy) is utilized to split the said document into individual sentences [37]. The set of all sentences in RV_i is now given by $E_i = \{e_{i1}, e_{i2}, ..., e_{ig}\}$ where g refers to the total number of sentences. Afterward, E_i is fed to a pre-trained BERT model to obtain their corresponding sentence embeddings. This process produces the set of sentence embeddings $E'_i = \{e'_{i1}, e'_{i2}, ..., e'_{ig}\}$, with $E'_i \in \mathbb{R}^{g \times 1024}$. The BERT model options can either be standard BERT-Large, wherein the contextualized embeddings can be based on the penultimate encoder layer [38] or Sentence-BERT [39], which is based on RoBERTa-Large prior trained on the semantic textual similarity task.

3.2.2 Embedding Clustering

Embedding clustering, based on K-Means, is next performed to partition the sentence embeddings in E'_i into K clusters. In our approach, K is a hyperparameter whose value may be selected using either manner:

- **Fixed-Value (FVK):** The value of K is manually set. This means that all items, regardless of their respective number of sentences in the review set, have the same amount of sentences in the extractive summary. In other words, all items have the same K.
- **Ratio-Based (RBK):** Given by the formula below, the value of K is determined by another hyperparameter called ϕ, which refers to the percentage of sentences that shall later comprise the summary. This implies that all items have the same ϕ but different K as each item has a different number of sentences (g) in the first place.

$$K = \phi \times g \tag{4}$$

The objective of embedding clustering is to minimize the Sum of Squared Errors (SSE), i.e., the intra-cluster sum of the distances from each sentence to its nearest centroid, given by the following equation [40]:

$$SSE_i = \sum_{x=1}^{K} \sum_{e'_{iy} \in C_x} ||e'_{iy} - c_x||^2 \tag{5}$$

where c_x is the centroid of cluster C_x that is closest to the sentence embedding e'_{iy}. The said objective function is optimized for item i by running the assignment and update steps until the cluster centroids stabilize. The assignment step assigns each sentence to a cluster based on the shortest distance between the sentence embedding and cluster centroid, provided by the formula below:

$$nc(e'_{iy}) = argmin_{x=1,...,K}\{||e'_{iy} - c_x||^2\} \tag{6}$$

where nc is a function that obtains the cluster closest to e'_{iy}. Then, the update step recomputes the cluster centroids based on new assignments from the previous step. This is defined as:

$$c_x = \frac{1}{|C_x|} \sum_{y=1}^{g} \{e'_{iy} | nc(e'_{iy}) = x\} \tag{7}$$

where $|C_x|$ refers to the number of sentences that cluster C_x contains. By introducing clustering, redundant and related sentences are grouped in the same cluster. Sentences closest to each cluster centroid are selected and combined to form the extractive summary. This is expressed as:

$$\begin{aligned} ns(C_x) &= argmin_{y=1,...,g}\{||e'_{iy} - c_x||^2\} \\ XS_i &= [e'_{i,ns(C_1)}, e'_{i,ns(C_2)}, ..., e'_{i,ns(C_K)}] \\ \overline{XS}_i &= \frac{1}{K} \sum_{x=1}^{K} e'_{i,ns(C_x)} \end{aligned} \tag{8}$$

where ns is a function that returns the nearest sentence to the centroid c_x of cluster C_x, $XS_i \in \mathbb{R}^{K \times 1024}$ is an embedding matrix of the extractive summary sentences, and $\overline{XS}_i \in \mathbb{R}^{1 \times 1024}$ is the extractive summary embedding of item i. Each sentence in the summary essentially represents a cluster.

3.2.3 Rating-Based Hidden Vectors and Fusion Layers

We also draw certain principles from the traditional latent factor model by incorporating hidden vectors that depict users and items to a certain extent [1]. These vectors, which

TABLE 3
Information about the datasets utilized in this study.

Dataset	#Reviews	#Users	#Items
Digital Music	64,706	5,541	3,568
Instant Video	37,126	5,130	1,685
Office Products	53,258	4,905	2,420
Patio, Lawn, & Garden	13,272	1,686	962

learn interactions based on rating data alone, are obtained from $P \in \mathbb{R}^{|U| \times m}$ and $Q \in \mathbb{R}^{|I| \times m}$, where P and Q respectively represent the user and item rating-based matrices, $|U|$ and $|I|$ refer to the total number of users and items, and m is the latent vector dimension. Similar to its original implementation [23], the weights of the embedding matrices are randomly initialized yet learned during the ESCOFILT module training.

The rating-based vectors for a given user u and item i are therefore given by P_u and Q_i correspondingly, both in $\mathbb{R}^{1 \times m}$. They are then fused with their corresponding extractive summary embeddings. This is facilitated by these fusion levels, illustrated by the following formulas:

$$\begin{aligned} f_u &= (\overline{XS}_u W_u + b_u) + P_u \\ f_i &= (\overline{XS}_i W_i + b_i) + Q_i \\ f_{ui} &= [f_u, f_i] \end{aligned} \tag{9}$$

where f_u and f_i pertain to the preliminary fusion layers and both are in $\mathbb{R}^{1 \times m}$; W_u and W_i are weight matrices in $\mathbb{R}^{1024 \times m}$; b_u and b_i refer to bias vectors; and $f_{ui} \in \mathbb{R}^{1 \times 2m}$ denotes the initial user-item interactions from the third fusion layer and is later fed to the MLP.

3.2.4 Multilayer Perceptron and Rating Prediction

The MLP is essential to model the collaborative filtering effect, i.e., to learn meaningful non-linear interactions between users and items. An MLP with multiple hidden layers implies a higher degree of non-linearity and flexibility. Similar to the strategy of He et al. [23], NEAR adopts an MLP with a tower pattern; the bottom layer is the widest while every succeeding top layer has fewer neurons. A tower structure enables the MLP to learn more abstract data features. Particularly, we halve the size of hidden units for each successive higher layer. NEAR's MLP component is defined as follows:

$$\begin{aligned} h_1 &= ReLU(f_{ui} W_1 + b_1) \\ h_L &= ReLU(h_{L-1} W_L + b_L) \end{aligned} \tag{10}$$

where h_L represents the L-th MLP layer, and W_L and b_L pertain to the L-th layer's weight matrix and bias vector, respectively. As far as the MLP's activation function is concerned, we select the rectified linear unit (ReLU), which yields better performance than other activation functions [23]. Finally, the MLP's output is projected to one more linear layer to produce the predicted rating:

$$\hat{r}_{ui} = h_L W_{L+1} + b_{L+1} \tag{11}$$

3.3 LGR: Label-Giving Regressor

While invariant explanations are generated by embedding clustering (an unsupervised learning task), variant explanations are to be generated by VET, a Transformer-based component of NEAR. However, training Transformer models for natural language generation will typically require target texts. In our case, this refers to actual ground-truth explanations that are unrealistic to be expected in every user-item pair of a large dataset. Therefore, inspired by the work of Elsahar et al. [41], we propose LGR, a clever, greedy approach to allow the self-supervision nature of variant explanation generation. Motivated by our non-usage of gold texts and datasets manually labeled with explanations, LGR is a supporting module for VET; it selects a particular section (i.e., a sentence) from the item's invariant explanation (i.e., a summary) that best indicates the rating for every user-item pair. The selected sentence then serves as the artificial ground-truth label. We argue that this greedy method ensures the *variance* of variant explanations to a degree.

In order to enable this task, separately fine-tuning a neural regression model (e.g., BERT) is a prerequisite. Every review (v_{ui}) in the dataset is fed to the said model, with its corresponding rating (r_{ui}) as the output. The regression task involved is akin to numeric sentiment analysis. For illustration purposes, the fine-tuned regression model is represented by a function called $regr$. Afterward, the user-item pair's artificial ground-truth is obtained; this pertains to the sentence whose $regr$-predicted numeric sentiment is closest to the ground-truth rating, which is mathematically expressed below:

$$lgr(u,i) = argmin_{x=1,...,K}\{||regr(e'_{i,ns(C_x)}) - r_{ui}||\} \tag{12}$$

where lgr returns the artificial gold explanation for a pair user u and item i.

3.4 VET: Variant Explanation Transformer

3.4.1 Input Sequence

For the training phase, VET's input is a heterogeneous sequence S consisting of the user ID u, item ID i, rating r_{ui}, and artificial ground-truth explanation $lgr(u,i)$. This is given by the following:

$$S = [u, i, r_{ui}, w_1, w_2, ..., w_{|lgr(u,i)|}] \tag{13}$$

where $w_1, w_2, ..., w_{|lgr(u,i)|}$ comprise the explanation's word sequence. The sequence's token representation is obtained by performing an embedding lookup based on token embeddings that are learned or fine-tuned by VET during its training. Regarding this, it should be noted that there are four matrices that correspond to four types of inputs in S. These matrices, which essentially serve as lookup tables, pertain to users, items, ratings, and words. They are respectively given by $UM \in \mathbb{R}^{|U| \times d}$, $IM \in \mathbb{R}^{|I| \times d}$, $RM \in \mathbb{R}^{5 \times d}$, and $WM \in \mathbb{R}^{|Z| \times d}$, where $|Z|$ is the vocabulary size, 5 refers to the highest possible rating that a user can give to an item, and d is the embedding dimension. As a quick example, encoding the rating of 3 is simply performed as a matrix lookup in RM, i.e., $RM_3 \in \mathbb{R}^{1 \times d}$.

Unlike other approaches, we emphasize that encoding the rating, along with user and item information, is necessary for better personalization. A user's preference for a

TABLE 4
Baselines' information summary.

Model	Recommender Type	Feature Extractor	Explanation Mechanism	Explanation Type
LightGCN	Interaction-Based	GCN	None	None
NCF	Interaction-Based	None	None	None
NGCF	Interaction-Based	GCN	None	None
BERT-CF	Review-Based	BERT	None	None
DeepCoNN	Review-Based	CNN	None	None
MPCN	Review-Based	Co-Attention	Co-Attention	Review-Level (Invariant)
NARRE	Review-Based	CNN	Vanilla Attention	Review-Level (Invariant)
NRT	Review-Based	None	RNN-Based Decoder	NLG-Derived (Variant)
PETER	Review-Based	Transformer	Transformer	NLG-Derived (Variant)
NEAR	Review-Based	BERT	Summarization & Customized Transformer	Summary-Level & NLG-Derived (Invariant & Variant)

personalized experience (i.e., in the form of variant explanations in this paper) can be attributed to his desire for control [19]. Once a user obtains something that is tailored to him, it makes him feel to be in control [42]. We argue that the rating can give us an approximate sense of what should be tailored to him. In other words, it serves as a greedy numerical indication of user interest in a specific item, thus, warranting the encoding of the rating data. Afterward, token positions in the sequence are encoded by utilizing positional embeddings. The positional representation is indicated by $[p_1, p_2, ..., p_{|S|}]$, where $|S|$ is the sequence length. Both the token and positional representations are then integrated in S_0 by obtaining their sum.

$$S_0 = [s_{01}, s_{02}, ..., s_{0|S|}] \tag{14}$$

3.4.2 Customized Transformer Configuration

NEAR's Transformer consists of L identical layers, and each layer has two mechanisms: the multi-headed self-attention and position-wise feedforward network. The current l-th layer (where $l \in [1, L]$) encodes the prior layer's output S_{l-1} into $S_l \in \mathbb{R}^{|S| \times d}$. Under the multi-headed self-attention mechanism, a given l-th encoder layer has H attention heads. To compute for the l-th layer's h-th head A_{lh}:

$$\begin{aligned} A_{lh} &= softmax(\frac{Q_{lh}K_{lh}^{\intercal}}{\sqrt{d}} + M)V_{lh} \\ Q_{lh} &= S_{l-1}W_{lh}^{Q} \\ K_{lh} &= S_{l-1}W_{lh}^{K} \\ V_{lh} &= S_{l-1}W_{lh}^{V} \\ M &= \begin{cases} 0, & \textit{Allowed to attend} \\ -\infty, & \textit{Disallowed to attend} \end{cases} \end{aligned} \tag{15}$$

where W_{lh}^{Q}, W_{lh}^{K}, and $W_{lh}^{V} \in \mathbb{R}^{d \times \frac{d}{H}}$ are projection matrices, and $M \in \mathbb{R}^{|S| \times |S|}$ is the attention masking matrix. This masking matrix dictates the manner that a token can attend to another token. In a typical generation task that uses a vanilla Transformer, M's content structure is as follows: its lower triangular part is fixed to 0, while the remaining portions are set to $-\infty$ that allows each token to attend only to past tokens including itself [12]. To ensure that explanation generation is effective, we introduce the rating-sensitive context prediction task to endow semantic meanings to users, items, and ratings alike. This necessitates the customization of the attention masking matrix by allowing the first three tokens, i.e., u, i, and r, to attend to each other, while the explanation tokens will still follow the standard left-to-right masking. The Transformer then produces this final sequence representation, given by the notation below:

$$S_L = [s_{L1}, s_{L2}, ..., s_{L|S|}] \tag{16}$$

The said representation is later projected to one more linear layer so as to allow its mapping to vocabulary Z:

$$c_t = softmax(W^z s_{Lt} + b^z) \tag{17}$$

where $W^z \in \mathbb{R}^{|Z| \times d}$ and $b^z \in \mathbb{R}^{|Z|}$ respectively refer to weight and bias parameters, and c_t is a vector that denotes the probability distribution over Z. The latter is employed to determine the probability c_t^w for word w.

3.4.3 Rating-Sensitive Context Prediction

Context prediction maps words to the user ID, item ID, and rating. It is the specific task that effectively grants them a semantic sense, instrumental in generating a more personalized explanation. Eliminating this task will lead the Transformer to merely produce identical sentences [12]. Context prediction's objective function is the negative log-likelihood (NLL) given by:

$$NLL_{cp} = \frac{1}{|\tau|} \sum_{u,i \in \tau} \frac{1}{|lgr(u,i)|} \sum_{t=1}^{|lgr(u,i)|} -log\ c_3^{w_t} \tag{18}$$

Examining the formula, the notation $c_3^{w_t}$ refers to the word w that is mapped to c_3 at time step t. The vector c_3 is the integrated representation of the first three inputs (i.e., user u, item i, and rating r_{ui}) since we adopt the masking method that permits them to attend to each other and, in effect, absorb each other's information. This task assigns an unordered list of words to every user-item-rating tuple. Yet, the sequence is unimportant in this instance because the main goal is the mapping of words, which is essentially an unordered task.

3.4.4 Variant Explanation Generation

The objective function adopted for this task is also the NLL:

$$NLL_{eg} = \frac{1}{|\tau|} \sum_{u,i \in \tau} \frac{1}{|lgr(u,i)|} \sum_{t=1}^{|lgr(u,i)|} -log\ c_{3+t}^{w_t} \tag{19}$$

Based on the equation, three positions, which correspond to the user, item, and rating representations, offset $c_t^{w_t}$ since the explanation text is located at the end of the sequence. Hence, this offset is expressed by the notation $c_{3+t}^{w_t}$.

During testing/inference, we highlight that there is no information about the rating r_{ui} that user u may accord item i. In this regard, we instead feed the rating approximated by ESCOFILT (rounded-off $\hat{r}_{ui}$) to the Transformer, along with u, i, and the begin-of-sequence token $<bos>$. VET predicts the next word based on a greedy algorithm that selects the word with the highest probability from the resulting probability distribution in $c_{<bos>}$. This step is repeated until either VET finds the end-of-sequence token $<eos>$ or the generated variant explanation VX_{ui} reaches a particular length. Finally, the explanation generation and rating-sensitive context prediction tasks are jointly trained to minimize the following loss function with respect to all trainable parameters of VET:

$$NLL = NLL_{cp} + NLL_{eg} \tag{20}$$

4 Empirical Evaluation

4.1 Datasets and Baselines

Table 3 summarizes the public Amazon datasets[1] utilized in our study. These datasets are said to be 5-core, wherein users and items are guaranteed to have at least five reviews each [43], [44]. The ratings across all datasets are in the range of [1, 5]. We split each dataset into training (80%), validation (10%), and test (10%) sets. To compare recommendation performance, listed below are the state-of-the-art baselines, whose information and other characteristics are outlined in Table 4.

- **LightGCN** [27]: A simplified version of the original NGCF model that utilizes a simple, linear approach of the weighted sum of embeddings for the user/item representations instead of NGCF's non-linear activation.
- **NCF** [23]: An interaction-based model that is fundamental in neural recommender systems and uses MLP as the interaction function for the first time.
- **NGCF** [25]: The first graph-based recommender model that employs collaborative signals during the embedding process by propagating embeddings on the bipartite user-item graph structure of interaction data.
- **BERT-CF**: We prepared a CF model that is solely based on two parallel pre-trained BERT models to learn user and item features, which are later fine-tuned on the rating prediction task.

1. http://jmcauley.ucsd.edu/data/amazon/

- **DeepCoNN** [24]: The first deep collaborative neural network model that is based on two parallel CNNs to jointly learn user and item features from reviews.
- **MPCN** [2]: Akin to NARRE, this CNN-less model employs a new type of dual attention, i.e., co-attention-based pointers, for identifying relevant reviews.
- **NARRE** [1]: Similar to DeepCoNN, it is a neural attentional regression model that integrates two parallel CNNs and the review-level attention mechanism.
- **NRT** [33]: The first model that can simultaneously predict ratings and generate explanations in the form of tips.
- **PETER** [12]: A unified model for the whole recommendation-explanation pipeline that is the first to enable the Transformer for generating personalized natural language explanations.

We also constructed two versions of NEAR, and their difference lies in setting the number of clusters in embedding clustering. **NEAR-FVK** adopts the fixed-value manner of selecting K, implying that the number of K clusters is the same for all user/item review sets. This is in direct contrast to **NEAR-RBK**, which employs the ratio-based approach of deciding K's value across all review sets.

Moreover, to differentiate personalized explanation performance, besides PETER and NRT, we employed the following:

- **RAND**: Provides a text of randomly generated tokens according to a pre-defined length.
- **NEAR-NR**: A variant of our NEAR model that does not encode the rating information. This implies that the input sequence of VET (in Eq. 13) is modified accordingly; it only consists of the user, item, and artificial ground-truth explanation tokens:

$$S = [u, i, w_1, w_2, ..., w_{|lgr(u,i)|}] \tag{21}$$

4.2 Evaluation Metrics

4.2.1 Recommendation Performance

The **Root Mean Square Error (RMSE)**, measured on the test dataset $\bar{\tau}$, is a widely used metric for evaluating a model's rating prediction accuracy [45].

$$RMSE = \sqrt{\frac{1}{|\bar{\tau}|} \sum_{u,i \in \bar{\tau}} (r_{ui} - \hat{r}_{ui})^2} \tag{22}$$

4.2.2 Personalized Explanation Performance

1) **Average Between-Methods Pairwise Similarity (ABPS)**
We introduce the ABPS score, which measures the relative personalization between any two explanation-generating methods, i.e., how much the explanation texts are similar/dissimilar between them. For every user-item pair (or data point), explanations from two methods are matched and compared by computing the cosine similarity between their corresponding sentence embeddings. A lower ABPS value suggests that the

TABLE 5
Recommendation performance comparison of the baselines across all datasets.
The best and second-best RMSE scores are respectively boldfaced and underlined.

Model	Digital Music	Instant Video	Office Products	Patio, Lawn, & Garden	Average RMSE
LightGCN	0.9522	1.0459	0.8749	1.0088	0.9704
NCF	1.0822	1.1088	1.0008	1.2359	1.1069
NGCF	0.9026	1.0158	0.8508	0.9852	0.9386
BERT-CF	0.9607	1.0138	0.8725	0.9677	0.9536
DeepCoNN	0.8904	0.9778	<u>0.8410</u>	<u>0.9316</u>	<u>0.9102</u>
MPCN	0.9298	0.9976	0.8487	0.9362	0.9280
NARRE	0.8915	0.9758	0.8426	0.9539	0.9159
NRT	1.1505	1.4366	0.9216	1.0613	1.1425
PETER	0.9445	1.0402	0.8999	0.9647	0.9623
NEAR-RBK	**0.8823**	<u>0.9742</u>	**0.8332**	**0.9298**	**0.9049**
NEAR-FVK	<u>0.8831</u>	**0.9732**	0.8463	0.9403	0.9107

two methods are more different from each other. Computing ABPS is given by the following equation:

$$ABPS(m_1, m_2) = \frac{1}{|\bar{\tau}|}\sum_{u,i \in \bar{\tau}} cos(VX_{ui}^{m_1}, VX_{ui}^{m_2}) \quad (23)$$

where m_1 and m_2 pertain to the explanation-generating methods being compared, and cos is a function that computes the cosine similarity.

2) **Rating-Explanation Disagreement (RED)**
Operating under the desire for control premise, we likewise propose the RED metric to measure how much the generated explanation agrees/disagrees with the rating it is supposed to reflect . This is mathematically indicated by:

$$RED = \frac{1}{|\bar{\tau}|}\sum_{u,i \in \bar{\tau}} |regr(VX_{ui}) - r_{ui}| \quad (24)$$

where $regr$ is a prior and separately trained regression model on every review-rating pair of the training dataset. The lower the RED score, the better; a higher RED value implies that the generated explanation does not agree with the rating. To observe fairness in our experiments, we trained two distinct regressors with varying review feature sources:

- **SVR** [46]: We extracted the reviews' TF-IDF features to serve as inputs of the support vector regression (SVR) model. SVR finds the best fit line on those features to predict the numeric sentiment.
- **DistilBERT** [47]: This is a fine-tuned neural regression model whose size is 40% of the original BERT model, while maintaining 97% of its language understanding capabilities and being 60% faster.

4.3 Experimental Settings

For NEAR's CF component, the number of MLP layers was set to 4. We operated an exhaustive grid search over the following hyperparameters:

- Learning rates: {0.004, 0.006}
- Number of epochs: [1, 30]
- Latent vector dimension (m): {128, 200, 220}
- Item K (for NEAR-FVK only): {3, 10, 20, 40}
- User K (for NEAR-FVK only): {3, 10, 20, 40}

For NEAR-RBK, the summary ratio ϕ was fixed to 40%. For LightGCN, NCF, and NGCF, we performed a grid search over the number of epochs [1, 30] and latent vector dimension {128, 220}. For NRT, we maintained the settings reported by Li et al.[2]. Moreover, for BERT-CF, DeepCoNN, MPCN, and NARRE, we employed the extensible NRRec[3] framework [48] and likewise adopted the other hyperparameters reported in the framework. We also performed an exhaustive grid search over the number of epochs [1, 30] and learning rates {0.003, 0.004, 0.006}. All these models mentioned above used the same optimizer, Adam, which leverages the power of adaptive learning rates during training [49]. We selected the model configuration (i.e., grid point) with the lowest RMSE on the validation set.

Furthermore, for PETER and NEAR's VET component, the following values were fixed:

- Maximum epoch: 100
- Maximum explanation length: 30
- Vocabulary size: 100 000
- Feedforward network dimension: 2048
- Initial learning rate: 1.0
- Embedding size (d): 512
- Number of layers (L): 2
- Number of attention heads (H): 2

These Transformer-based approaches employed the same optimizer, stochastic gradient descent (SGD), and applied gradient clipping with 1.0 as the threshold. They used early stopping such that when no improvement was observed, the learning rate would be decreased by a factor of 0.25. When this occurred five times, training was then terminated.

2. https://github.com/lileipisces/NRT
3. https://github.com/ShomyLiu/Neu-Review-Rec

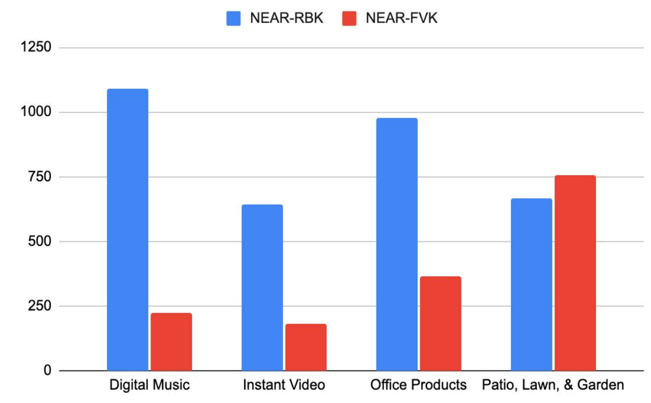

Fig. 2. Average number of words used to form the item embedding and extractive summary by NEAR's best performing grid point on the validation set.

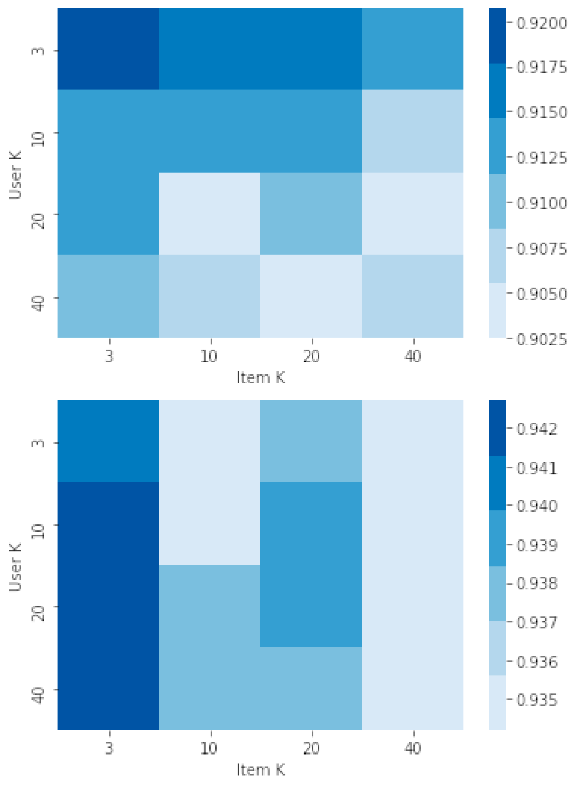

Fig. 3. Illustration of NEAR's performance by varying the values of user K and item K for the Digital Music (top heatmap) and Patio, Lawn, & Garden (bottom heatmap) datasets.

5 Results and Discussion

5.1 Recommendation Performance

Table 5 reports the RMSE results of our experiments. These are our general findings and observations:

First, NEAR is the top-performing recommender model, having obtained the lowest RMSE values across all datasets and state-of-the-art baselines. This finding affirms the effectiveness of NEAR's CF component rooted in the successful integration of BERT, embedding clustering, and rating-based hidden vectors for both user and item modeling networks. Employing a BERT-influenced network rather

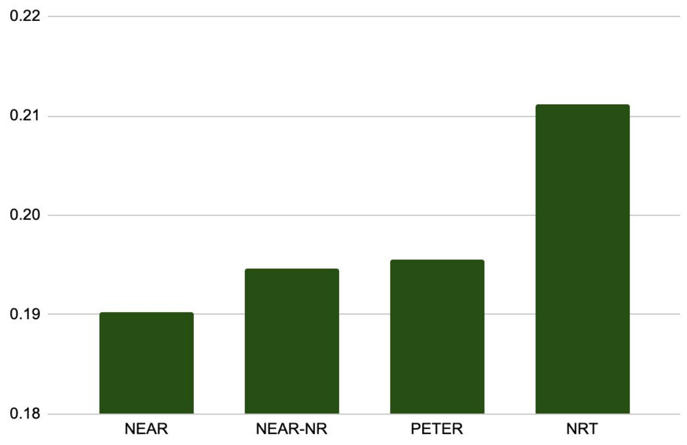

Fig. 4. Pairwise similarity comparison between the random baseline (RAND) and each of the explanation-generating baselines in terms of ABPS.

TABLE 6
Rating-explanation disagreement comparison. RED_{SVR} and RED_{DB} denote the RED scores calculated by SVR and DistilBERT, respectively. RED_{μ} is the mean of the two RED values. The best values are boldfaced.

Model	$\mathbf{RED_{SVR}}$	$\mathbf{RED_{DB}}$	$\mathbf{RED_{\mu}}$
NEAR	**0.6802**	**0.4604**	**0.5703**
NEAR-NR	0.7628	0.7252	0.7440
NRT	0.9460	0.8234	0.8847
PETER	0.7582	0.7084	0.7333
RAND	1.1127	0.8339	0.9732

than CNN as the review feature extractor results in excellent rating prediction accuracy. Nevertheless, it is worth noting that using vanilla BERT alone is insufficient, as evidenced by the BERT-CF's relatively worse RMSE scores among review-based recommenders. This signifies that the full benefits of BERT may only be realized when integrated with other relevant neural components.

Second, models that exploit review information (i.e., NEAR, DeepCoNN, NARRE, and MPCN) are the four best recommenders, consistently outperforming models solely based on ratings or interactions (i.e., LightGCN, NGCF, and NCF). Though propagating the bipartite graph of user-item interactions in embeddings has been beneficial (as both LightGCN and NGCF have better RMSEs than the regular NCF), review texts are still far better sources of rich latent information for learning user and item characteristics when adequately employed. Generally, review-based recommender systems are considered reliable in yielding satisfactory and quality recommendation performance.

Third, PETER and NRT produce the worst RMSE scores among all review-based recommender models. It should be first noted that in PETER's case, more premium is placed on personalized explanation generation than accurate recommendation. These results indicate that jointly training the rating prediction and explanation generation tasks negatively affects the accuracy of the former. Thus, separately

TABLE 7
Pairwise similarity matrix among Transformer-based methods (in terms of ABPS). The lowest values are boldfaced.

Model	NEAR	NEAR-NR	PETER
NEAR	N/A	**0.6730**	**0.6860**
NEAR-NR	**0.6730**	N/A	0.8447
PETER	**0.6860**	0.8447	N/A

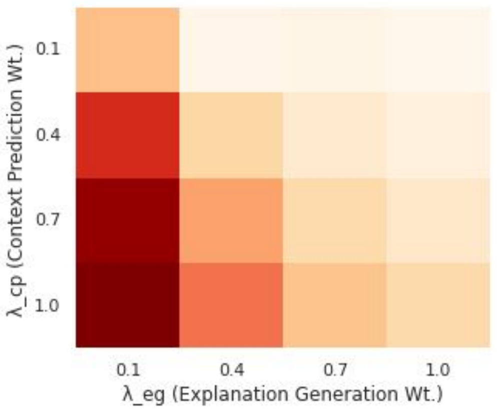

Fig. 5. Illustration of the pairwise similarity (ABPS) between the random baseline (RAND) and each weighted version of NEAR's VET. A darker color indicates a lower ABPS value, implying that the VET version produces explanations that are less semblant to randomly generated tokens.

training a CF model for rating prediction alone still appears to be the optimal choice.

Fourth, the performance of NEAR-FVK is generally lower than NEAR-RBK. NEAR-FVK's dip in performance may be attributed to fewer clusters used to form user/item representations in the embedding clustering procedure. Fewer clusters then imply fewer sentences/words and less extractable information for generalization during training. For example, NEAR-RBK employs 60 clusters on average, while NEAR-FVK uses 10 clusters (regardless of the review's length) to obtain the Digital Music dataset's optimal RMSE. To further support our point, Fig. 2 illustrates the average number of words used by NEAR-RBK and NEAR-FVK to form the item embedding and extractive summary; the latter being significantly less than the former. Take note that for the first three datasets, the optimal value of NEAR-FVK's item K is 10. For the Patio dataset, its best value is 40; hence, the average number of words for both NEAR-RBK and NEAR-FVK is nearly the same.

Overall, NEAR-RBK has better generalization capability and more stability than NEAR-FVK. Yet, we argue that NEAR-FVK is as at least competitive as the other state-of-the-art baselines. The said model can be used to approximate NEAR-RBK's performance, being an acceptable compromise approach when certain factors are considered such as presentation of recommendations and management of computing resources. Specifically, setting the number of clusters/sentences to a fixed value for the extractive summary helps deal with information overload among users. It may also have some implications on the better utilization of limited resources (e.g., GPU memory).

5.1.1 *Analysis on the Homogeneity and Heterogeneity of Features*

Another key purpose of preparing the fixed-value K version of NEAR is to analyze the effect of adjusting the values of user K and item K as far as the homogeneity and heterogeneity of features are concerned. Different K combinations are explored in Fig. 3, which illustrates the heatmaps for the Digital Music and Patio, Lawn, & Garden datasets. The lightest blue shades depict the optimal results in the figure. A notable observation for both heatmaps is that setting item K (shown in the x-axis) to a lower value (such as 3) results in poor performance while adjusting it to a higher value (e.g., 40) generally gives us the lightest shades. These findings validate the heterogeneity of item features. On the other hand, user features are more homogeneous than item features. When item K is already set to a proper value (e.g., 40), fixing user K to any value generally results in a respectable performance. Hence, the adjustment of item K is more crucial than user K in affecting NEAR's predictive performance.

5.2 Variant Explanation Analysis

5.2.1 *Personalization Analysis*

Explanation-generating approaches can be classified into two types: the model that encodes user, item, and rating information (i.e., NEAR) and models that only encode user and item information (i.e., NEAR-NR, NRT, and PETER).

We first calculate the pairwise similarity between the random baseline and each of the explanation-generating methods. As depicted in Fig. 4, NEAR has the lowest (best) RAND-similarity value. This is followed by PETER and NEAR-NR, and NRT has the highest (worst) RAND-similarity value. These findings suggest that without leveraging the rating information, a model may provide explanation texts that are less personalized and more semblant to random tokens.

Table 7 shows the pairwise similarity matrix among the Transformer-based explanation-generating approaches. Our NEAR model is evidently the most dissimilar; its similarity values with NEAR-NR and PETER are 0.6730 and 0.6830, respectively. On the contrary, the similarity between NEAR-NR and PETER is 0.8447, a considerably higher value. These results imply that leveraging rating information enhances the personalization and diversity of explanation texts.

Lastly, we examine a specific example whose attention visualization is illustrated in Fig. 7. Whenever NEAR generates a variant explanation, information from the user, item, and rating can varyingly affect it. It is worth mentioning that the rating information carries a meaningful influence on the overall generation process. The predicted rating of 4 (out of 5, the highest rating) grants the most attention to these tokens: {*"It"*, *"is"*, *"not"*, *"bad"*}, effectively personalizing the text according to the rating-sentiment.

Overall, the observations in this section give credence to our approach of conditioning the Transformer not only on every user-item pair but also on every user-item-rating tuple, leading to better personalization performance.

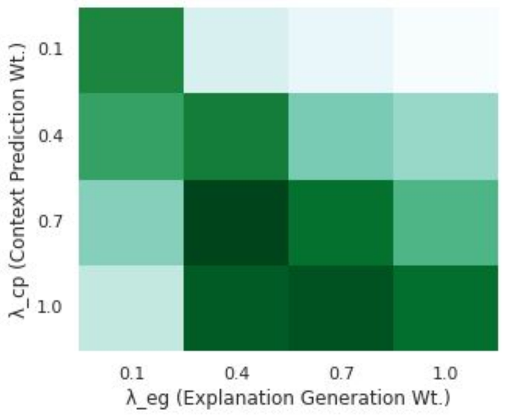

Fig. 6. Illustration of the RED_{DB} scores of the weighted versions of NEAR's VET. A darker color indicates a lower RED score, meaning that the version's generated explanations generally agree with the ground-truth ratings.

5.2.2 Rating-Explanation Disagreement Results

Table 6 outlines the calculated RED scores for NEAR and the rest of the baselines. NEAR has the best RED values for both SVR- and DistilBERT-based regressors, and it is followed far behind by NEAR-NR and PETER. The worst results come from NRT and the random baseline (RAND). These suggest that our manner of incorporating the rating information to our customized Transformer effectively produces explanations that reflect/agree with their corresponding ratings. Therefore, the original NEAR model has the most promise when it comes to dealing with the user's desire for control.

5.2.3 Objective Functions' Influence to NEAR's Personalized Explanation Performance

As previously seen in Eq. 20, the following equation expresses the combined objective function (NLL) to train NEAR's VET component on the tasks of variant explanation generation (NLL_{eg}) and rating-sensitive context prediction (NLL_{cp}):

$$NLL = NLL_{cp} + NLL_{eg} \tag{25}$$

To examine the respective influences of these two tasks on the personalization of explanations, we modify the original objective function to a weighted version. This is then given by:

$$NLL = \lambda_{cp} NLL_{cp} + \lambda_{eg} NLL_{eg} \tag{26}$$

where λ_{cp} and λ_{eg} are weights for the individual loss functions of the two tasks. The possible values of λ_{cp} and λ_{eg} are in $\{0.1, 0.4, 0.7, 1.0\}$ for the purposes of our experiments. With a grid search-like computation, we essentially have 16 weighted versions of NEAR's VET. Afterward, we compute the pairwise similarity between each version and the random baseline ($ABPS(NEAR^{\lambda_{eg}}_{\lambda_{cp}}, RAND)$) and RED score of the former (specifically, RED_{DB} of $NEAR^{\lambda_{eg}}_{\lambda_{cp}}$).

Based on Fig. 5 and Fig. 6, there seems to be a trend; the darkest shades take the form of a lower left triangle in both figures, indicating lower RAND-pairwise similarity values and RED scores. Optimal results (i.e., explanations more dissimilar to random tokens and more indicative of the rating) are generally achieved if $\lambda_{cp} >= \lambda_{eg}$. In other words, this expresses that the influence of rating-sensitive context prediction is greater than or equal to variant explanation generation in personalizing explanation texts.

5.3 Invariant Explanation Analysis

This section involves performing manual assessments of explanation texts. Considering this, we hired the services of four human judges for each of our evaluation schemes. We pre-screened them according to these characteristics of a potential end-user: online shopping frequency is multiple times per month, and the first or primary language is English. The former is particularly important since the most common application domain of recommender systems is e-commerce [50]. The latter qualification removes any biases or untoward results that may be caused by a lack of English fluency. Additionally, we examined the assessment results and determined the strength of inter-judge agreement by using Fleiss' Kappa (κ) wherein -1 indicates a less than chance agreement, 0 refers to a random agreement, and 1 denotes a perfect agreement [51]. We also performed a grid search-like computation of the Kappa coefficient for every possible set/subset of judges.

5.3.1 Assessment on Embedding Clustering

We explore the effectiveness of embedding clustering by determining whether the invariant explanations' clusters can overall distinguish aspects, attributes, or themes. We asked the human judges to identify such aspects in 20 randomly selected clusters, and each cluster contains about 10 sentences; this amounts to a total of 200 texts to be evaluated. We specifically instructed them to rate the level of difficulty of performing aspect identification on a five-point Likert scale: 1 signifies *very difficult* and 5 denotes *very easy*. Our rationale behind this assessment strategy is that if an attribute cannot be easily identified, this implies that a given cluster (also by extension, embedding clustering) fails to distinguish the underlying aspect. According to Fig. 8, the human judges overwhelmingly found it at least easy to identify aspects/properties for about 64% of the time (37.5% for very easy and 26.25% for easy). On certain occasions, they thought that the task was at least difficult (17.5%) and neither easy nor difficult (18.75%). The ranking consistency Kappa values for two subsets of raters lie in the range of [0.11, 0.21], signaling their agreement to a certain degree. These findings confirm the efficacy of our approach in producing attribute-distinguishing clusters.

5.3.2 Assessment on Real-Life Explainability

Unlike a variant explanation based on a single tuple of information (making automatic evaluation metrics a natural fit), an invariant explanation is derived from multiple tuples of information, thus complicating the challenges of its evaluation. Previous evaluation tools depended on simple qualitative analysis and visualization of attention weights on selected samples [2], [11], [14], [16], [29]. Nevertheless, to the best of our knowledge, there does not appear to be a comprehensive set of criteria that evaluates the real-life acceptability of explanations. Regarding this, we propose the following explainability criteria, which are inspired by Zemla et al. [52].

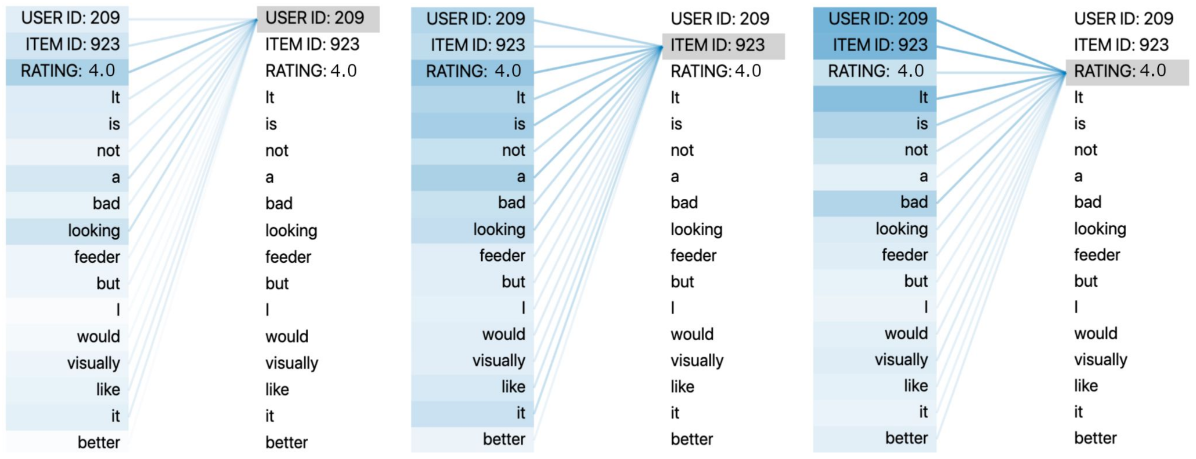

Fig. 7. Attention visualization of the respective influences of user, item, and rating information on the explanation generation process. Darker blue shades indicate higher attention.

TABLE 8
Assessment of invariant explanations based on the real-life explainability criteria (pointwise evaluation). The best mean values for each criterion are boldfaced.

Explanation Type	Coherence	Completeness	Lack of Alternatives	Novelty	Perceived Truth	Quality	Visualization
Review-Level	3.66	3.29	**3.18**	**3.43**	**3.80**	3.79	3.58
Token-Level	3.46	3.03	2.66	2.89	3.75	3.36	3.19
Summary-Level	**3.69**	**3.34**	2.96	3.28	3.73	**3.83**	**3.60**

TABLE 9
Ranking-based assessment of invariant explanations (listwise evaluation). *The %ranked N-th* columns indicate the distribution (in %) of the rankings given by the human judges.

Explanation Type	%Ranked 1st	%Ranked 2nd	%Ranked 3rd
Review-Level	35.00%	51.25%	13.75%
Token-Level	0.00%	21.25%	78.75%
Summary-Level	65.00%	27.50%	7.50%

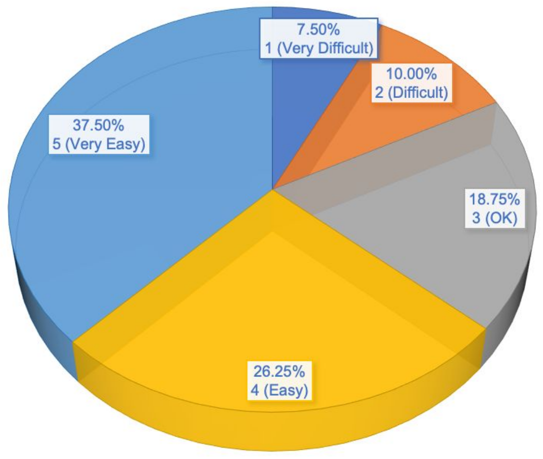

Fig. 8. Distribution of the judges' given difficulty level in identifying cluster aspects.

1) **Coherence:** "Parts of the explanation fit together coherently."
2) **Completeness:** "There are no gaps in the explanation."
3) **Lack of Alternatives:** "There are probably less to no reasonable alternative explanations."
4) **Novelty:** "I learned something new from the explanation."
5) **Perceived Truth:** "I believe this explanation to be true."
6) **Quality:** "This is a good explanation."
7) **Visualization:** "It is easy to visualize what the explanation is saying."

For the assessment, we produced a total of 60 invariant explanations, 20 per explanation type: review-level (NARRE), token-level (BENEFICT), and summary-level (NEAR). In pointwise evaluation, we asked the human judges to assess them using our proposed real-life explainability criteria on a five-point Likert scale. In listwise evaluation, we instructed the raters to rank the explanations according to helpfulness. A statement is considered helpful if it can aid the user both to know a product better and make a potential purchasing decision.

Table 8 shows the results of pointwise evaluation wherein summary-level explanations are on par with or better than review-level explanations. Our summary-level explanations dominate four out of seven criteria. They are found to be most coherent and most complete. They also have the best quality and are the easiest to visualize. Their strongest aspect is quality, acquiring a mean rating of 3.83. This is later affirmed by Kappa values in the range of [0.10, 0.20] from two pairs of judges, indicating their agreement to a certain extent. Furthermore, review-level explanations have the highest scores in lack of alternatives, novelty, and perceived truth. Their best aspect is novelty, with a mean rating of 3.43 and $\kappa = 0.28$ from a pair of judges, also describing a fair agreement. Likewise, Table 9 presents the listwise evaluation results that illustrate the overwhelming preference for summary-level explanations. According to the table, they are ranked first 65% of the time. These are followed far behind by review-level explanations, ranked first for 35% of the items. On the other hand, token-level explanations are never ranked first while being deemed third for nearly 79%. The calculated Fleiss' Kappa for overall ranking consistency is about 0.30, indicating a fair amount of agreement among all judges.

6 Conclusion

Our paper successfully demonstrates the development of a complete CF framework that can generate non-supervised explanations, i.e., unsupervised invariant (summary-level) and self-supervised variant texts (NLG-derived), while maintaining outstanding rating prediction accuracy. Our experiments and assessments show that NEAR-generated variant explanations are more personalized than those from other baseline models, and invariant explanations are preferred over other contemporary models' texts in real life. We argue that our study offers an avenue to help bridge the gap between accuracy and explainability, two crucial aspects that our proposed architecture equally emphasizes.

Acknowledgment

This work was funded in part by Qualcomm through a Taiwan University Research Collaboration Project and also in part by the Ministry of Science and Technology, Taiwan, under NCKU B109-K027D and MOST 109-2221-E-006-173 grants, respectively.

References

[1] C. Chen, M. Zhang, Y. Liu, and S. Ma, "Neural attentional rating regression with review-level explanations," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 1583–1592.

[2] Y. Tay, A. T. Luu, and S. C. Hui, "Multi-pointer co-attention networks for recommendation," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2018, pp. 2309–2318.

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

[4] C. Musto, M. de Gemmis, G. Semeraro, and P. Lops, "A multi-criteria recommender system exploiting aspect-based sentiment analysis of users' reviews," in *Proceedings of the 11th ACM Conference on Recommender Systems*, 2017, pp. 321–325.

[5] Z. Chen, X. Wang, X. Xie, T. Wu, G. Bu, Y. Wang, and E. Chen, "Co-attentive multi-task learning for explainable recommendation." in *IJCAI*, 2019, pp. 2137–2143.

[6] 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*, 2018, pp. 2060–2069.

[7] 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 & Data Mining*, 2016, pp. 1135–1144.

[8] X. Wang, X. He, F. Feng, L. Nie, and T.-S. Chua, "TEM: Tree-enhanced embedding model for explainable recommendation," in *Proceedings of the 2018 World Wide Web Conference*, 2018, pp. 1543–1552.

[9] Y. Zhang, G. Lai, M. Zhang, Y. Zhang, Y. Liu, and S. Ma, "Explicit factor models for explainable recommendation based on phrase-level sentiment analysis," in *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2014, pp. 83–92.

[10] Y. Zhang and X. Chen, "Explainable recommendation: A survey and new perspectives," *arXiv preprint arXiv:1804.11192*, 2018.

[11] X. Chen, Y. Zhang, and Z. Qin, "Dynamic explainable recommendation based on neural attentive models," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 53–60.

[12] L. Li, Y. Zhang, and L. Chen, "Personalized transformer for explainable recommendation," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Aug. 2021, pp. 4947–4957.

[13] X. Feng and Y. Zeng, "Neural collaborative embedding from reviews for recommendation," *IEEE Access*, vol. 7, pp. 103 263–103 274, 2019.

[14] D. Liu, J. Li, B. Du, J. Chang, and R. Gao, "DAML: Dual attention mutual learning between ratings and reviews for item recommendation," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 344–352.

[15] R. A. Pugoy and H.-Y. Kao, "BERT-based neural collaborative filtering and fixed-length contiguous tokens explanation," in *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, 2020, pp. 143–153.

[16] S. Seo, J. Huang, H. Yang, and Y. Liu, "Interpretable convolutional neural networks with dual local and global attention for review rating prediction," in *Proceedings of the 11th ACM Conference on Recommender Systems*, 2017, pp. 297–305.

[17] R. A. Pugoy and H.-Y. Kao, "Unsupervised extractive summarization-based representations for accurate and explainable collaborative filtering," in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, Aug. 2021, pp. 2981–2990.

[18] L. Li, Y. Zhang, and L. Chen, "Generate neural template explanations for recommendation," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 755–764.

[19] A. Dillon, "Collecting as routine human behavior: Personal identity and control in the material and digital world," *Information & Culture*, vol. 54, no. 3, pp. 255–280, 2019.

[20] A. Faraji-Rad, S. Melumad, and G. V. Johar, "Consumer desire for control as a barrier to new product adoption," *Journal of Consumer Psychology*, vol. 27, no. 3, pp. 347–354, 2017.

[21] L. Wu, X. He, X. Wang, K. Zhang, and M. Wang, "A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation," *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[22] X. He, X. Du, X. Wang, F. Tian, J. Tang, and T.-S. Chua, "Outer product-based neural collaborative filtering," *arXiv preprint arXiv:1808.03912*, 2018.

[23] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web*, 2017, pp. 173–182.

[24] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proceedings of the 10th ACM International Conference on Web Search and Data Mining*, 2017, pp. 425–434.

[25] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval*, 2019, pp. 165–174.

[26] Z. Liu, L. Meng, F. Jiang, J. Zhang, and P. S. Yu, "Deoscillated graph collaborative filtering," *arXiv preprint arXiv:2011.02100*, 2020.

[27] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, and M. Wang, "Lightgcn: Simplifying and powering graph convolution network for recommendation," in *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, 2020, pp. 639–648.

[28] X. Dong, J. Ni, W. Cheng, Z. Chen, B. Zong, D. Song, Y. Liu, H. Chen, and G. de Melo, "Asymmetrical hierarchical networks with attentive interactions for interpretable review-based recommendation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 05, 2020, pp. 7667–7674.

[29] C. Wu, F. Wu, J. Liu, and Y. Huang, "Hierarchical user and item representation with three-tier attention for recommendation," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 1818–1826.

[30] M. T. Pilehvar and J. Camacho-Collados, "WiC: the word-in-context dataset for evaluating context-sensitive meaning representations," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, 2019, pp. 1267–1273.

[31] Q. Wang, S. Li, and G. Chen, "Word-driven and context-aware review modeling for recommendation," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1859–1862.

[32] L. Dong, S. Huang, F. Wei, M. Lapata, M. Zhou, and K. Xu, "Learning to generate product reviews from attributes," in *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, 2017, pp. 623–632.

[33] 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*, 2017, pp. 345–354.

[34] G. Zhao, H. Fu, R. Song, T. Sakai, Z. Chen, X. Xie, and X. Qian, "Personalized reason generation for explainable song recommendation," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 10, no. 4, pp. 1–21, 2019.

[35] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[36] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[37] S. Gupta and K. Nishu, "Mapping local news coverage: Precise location extraction in textual news content using fine-tuned bert based language model," in *Proceedings of the 4th Workshop on Natural Language Processing and Computational Social Science*, 2020, pp. 155–162.

[38] D. Miller, "Leveraging BERT for extractive text summarization on lectures," *arXiv preprint arXiv:1906.04165*, 2019.

[39] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[40] S. Xia, D. Peng, D. Meng, C. Zhang, G. Wang, E. Giem, W. Wei, and Z. Chen, "A fast adaptive k-means with no bounds," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2020.

[41] H. Elsahar, M. Coavoux, M. Gallé, and J. Rozen, "Self-supervised and controlled multi-document opinion summarization," *arXiv preprint arXiv:2004.14754*, 2020.

[42] L. F. Bright, *Consumer control and customization in online environments: an investigation into the psychology of consumer choice and its impact on media enjoyment, attitude and behavioral intention*. The University of Texas at Austin, 2008.

[43] J. McAuley, C. Targett, Q. Shi, and A. Van Den Hengel, "Image-based recommendations on styles and substitutes," in *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2015, pp. 43–52.

[44] 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*, 2016, pp. 507–517.

[45] H. Steck, "Evaluation of recommendations: rating-prediction and ranking," in *Proceedings of the 7th ACM Conference on Recommender systems*, 2013, pp. 213–220.

[46] M. Awad and R. Khanna, "Support vector regression," in *Efficient learning machines*. Springer, 2015, pp. 67–80.

[47] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

[48] H. Liu, F. Wu, W. Wang, X. Wang, P. Jiao, C. Wu, and X. Xie, "NRPA: Neural recommendation with personalized attention," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2019, pp. 1233–1236.

[49] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[50] S. Sivapalan, A. Sadeghian, H. Rahnama, and A. M. Madni, "Recommender systems in e-commerce," in *2014 World Automation Congress (WAC)*. IEEE, 2014, pp. 179–184.

[51] R. M. Borromeo and M. Toyama, "Automatic vs. crowdsourced sentiment analysis," in *Proceedings of the 19th International Database Engineering & Applications Symposium*, 2015, pp. 90–95.

[52] J. C. Zemla, S. Sloman, C. Bechlivanidis, and D. A. Lagnado, "Evaluating everyday explanations," *Psychonomic Bulletin & Review*, vol. 24, no. 5, pp. 1488–1500, 2017.

Reinald Adrian Pugoy (Student Member, IEEE) received his BS and MS degrees in Computer Science from the University of the Philippines Los Baños in 2008 and 2011, respectively. He is currently a PhD candidate at the Department of Computer Science and Information Engineering (CSIE), National Cheng Kung University (NCKU), Taiwan. He is also with the Faculty of Information and Communication Studies, University of the Philippines Open University. His research interests include recommender systems, natural language processing (NLP), machine learning, and data mining.

Hung-Yu Kao (Member, IEEE) received his BS and MS degrees in Computer Science from the National Tsing Hua University, Taiwan in 1994 and 1996, respectively. In 2003, he received his PhD degree from the Electrical Engineering Department, National Taiwan University. He is currently a professor at the CSIE department of NCKU. Dr. Kao is also the chair of IEEE CIS Tainan Chapter. He was a postdoctoral fellow at the Institute of Information Science, Academia Sinica, Taiwan from 2003 to 2004. His research interests include NLP, web information retrieval and extraction, knowledge management, data mining, social network analysis, and bioinformatics.