Personalized Abstractive Opinion Tagging

Mengxue Zhao*
Shandong University
keninazhao@163.com

Yang Yang
Meituan
yangyang113@meituan.com

Miao Li
University of Melbourne
miao4@student.unimelb.edu.au

Jingang Wang†
Meituan
wangjingang02@meituan.com

Wei Wu
Meituan
wuwei19850318@gmail.com

Pengjie Ren
Shandong University
renpengjie@sdu.edu.cn

Maarten de Rijke
University of Amsterdam
m.derijke@uva.nl

Zhaochun Ren†
Shandong University
zhaochun.ren@sdu.edu.cn

ABSTRACT

An opinion tag is a sequence of words on a specific aspect of a product or service. Opinion tags reflect key characteristics of product reviews and help users quickly understand their content in e-commerce portals. The task of abstractive opinion tagging has previously been proposed to automatically generate a ranked list of opinion tags for a given review. However, current models for opinion tagging are not personalized, even though personalization is an essential ingredient of engaging user interactions, especially in e-commerce. In this paper, we focus on the task of personalized abstractive opinion tagging. There are two main challenges when developing models for the end-to-end generation of personalized opinion tags: sparseness of reviews and difficulty to integrate multi-type signals, i.e., explicit review signals and implicit behavioral signals. To address these challenges, we propose an end-to-end model, named POT, that consists of three main components: (1) a *review-based explicit preference tracker* component based on a hierarchical heterogeneous review graph to track user preferences from reviews; (2) a *behavior-based implicit preference tracker* component using a heterogeneous behavior graph to track the user preferences from implicit behaviors; and (3) a *personalized rank-aware tagging* component to generate a ranked sequence of personalized opinion tags. In our experiments, we evaluate POT on a real-world dataset collected from e-commerce platforms and the results demonstrate that it significantly outperforms strong baselines.

CCS CONCEPTS

• **Information systems** → **Summarization**; **Sentiment analysis**.

*Work performed during an internship at Meituan.
†Zhaochun Ren and Jingang Wang are the corresponding authors.

SIGIR '22, July 11–15, 2022, Madrid, Spain

ACM ISBN 978-1-4503-8732-3/22/07…$15.00
https://doi.org/10.1145/3477495.3532037

KEYWORDS

Review analysis; Abstractive summarization; E-commerce; Personalization

ACM Reference Format:
Mengxue Zhao, Yang Yang, Miao Li, Jingang Wang, Wei Wu, Pengjie Ren, Maarten de Rijke and Zhaochun Ren. 2022. Personalized Abstractive Opinion Tagging. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3477495.3532037

1 INTRODUCTION

As large volumes of reviews of products and services are published every day, users of e-commerce platforms need tools to make sense of those reviews. In order to help users quickly understand the key information of products to facilitate their choices, many e-commerce platforms provide *tips* and *aspects* to reflect the key characteristics of product reviews. Often, a salient sentence is extracted as a *tip* to provide an overview of product information [24, 25]. *Aspects* refer to manually defined categories of products; they have also been applied to label reviews in e-commerce platforms [2]. However, the information contained in tips and aspects cannot give users a comprehensive and diverse representation of products (see Fig. 1), which reduces the user consumption experience [26]. Several e-commerce platforms, such as Dianping[1] and Taobao[2], have begun to provide *opinion tags* that reflect user preferences. An opinion tag is a sequence of words on a specific aspect to describe a product or service. Fig. 1 lists an example of reviews, tips, aspects, and opinion tags from Dianping, the largest crowd-sourced e-commerce review platform. In contrast with tips and aspects, opinion tags are able to provide more comprehensive user-preferred information.

Manually creating opinion tags is time-consuming and labor-intensive. In order to automatically provide opinion tags based on product reviews, a number of methods have been proposed [7, 24, 26, 27, 43, 44]. Most methods only focus on extracting opinion tags from reviews while ignoring the ranking of different opinion tags. Li et al. [26] propose abstractive opinion tagging (AOT) to generate a ranked list of opinion tags from the product reviews. Unfortunately, AOT fails to provide personalized product information.

[1]https://www.dianping.com/
[2]https://www.taobao.com/

Brief information

★ **Title:** FLY PIZZA & HOODADAK CHICKEN
★ **Tip:** The dip is an exclusive secret recipe.
★ **Aspects:** Cost-effective, Service, Taste, Environment, Space …

Reviews

R_1: Delicious as always …The waiter is too handsome, and the service attitude is very good …I will definitely come more often in the future.
R_2: The location of the restaurant is very easy to find, and the space is very large …You can totally trust the level of service in this restaurant.
R_3: Overall cool experience …I was pleasantly surprised by the deliciousness of the dipping sauce…
R_4: The beef is very tender …waiter introduce the dishes attentively …
R_5: Totally overwhelmed by the deliciousness here, but the environment can be a bit noisy …

Opinion tags

U_1: Fairly quick and polite service, great value for money.
U_2: Recommend everyone to come! Excellent dipping sauce, and the space is quite spacious.

Figure 1: An example of a popular restaurant from Dianping.

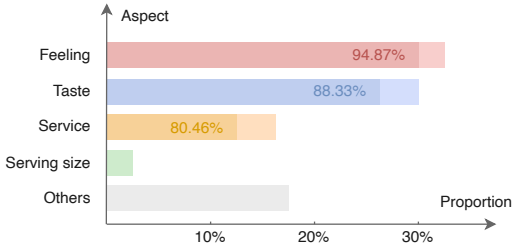

Figure 2: We count the top-3 aspects mentioned in each user's historical reviews, and show the proportion of the different aspects. Inside each bar, we also present the proportion of users who include the preferred aspect in recent reviews.

In this paper, we focus on generating opinion tags in a personalized way. However, establishing accurate mappings between user preferences and opinion tags is challenging. As an opinion tag is focused on one aspect, we consider using those predefined aspects as the bridge to connect users and opinion tags. Fig. 2 shows a distribution of aspects mentioned in reviews on Dianping. According to the distribution of aspects that users prefer, we find that users have diverse but consistent preferences on a set of aspects. For example, 94.87% users preferring "feeling" in historical reviews mention the same aspect again in their recent reviews. By leveraging user-preferred aspects to acquire personalization, we propose the task of *personalized abstractive opinion tagging*: we select the information that users are interested in from reviews, and then generate a ranked list of aspect and opinion tag pairs. As only a limited number of users provide reviews after consumption [18], it is difficult to capture inactive user interests purely from reviews. Inspired by previous studies on information retrieval and recommendation [12, 19, 29], we consider tracking user preferences not only using explicit feedback, i.e., reviews, but also using implicit behavior, e.g., clicks and purchases, etc. Two main challenges exist in personalized abstractive opinion tagging: (1) Sparseness of explicit feedback makes it difficult to explore preferences. (2) It is difficult to blend user implicit behaviors and explicit feedbacks.

To tackle the challenges listed above, we propose an end-to-end model, namely **POT**, which consists of three main components: (1) *review-based explicit preference tracker* (REPT), (2) *behavior-based implicit preference tracker* (BIPT), and (3) *personalized rank-aware tagging* (PRT). To tackle the sparseness in reviews, we propose a REPT based on a hierarchical heterogeneous review graph (HHRG) established by jointly modeling users, products, and reviews, to enrich explicit preferences. Based on a heterogeneous graph attention network, we track user preferences in HHRG by learning review-based representations with neighborhood features as supervision signals. We propose a BIPT to integrate implicit behavior and explicit feedback. BIPT tracks user preferences from a variety of implicit behavior based on a heterogeneous behavior graph (HBG). HBG mines potential user relations, and supplements them into user explicit preferences based on a multi-type behavior graph neural network. Finally, we propose a PRT to generate a ranked sequence of personalized aspect and opinion tag pairs.

Personalized opinion tagging data is difficult to obtain. As far as we know, there is no benchmark dataset supporting our task with comprehensive, diverse and personalized opinion tags. To evaluate the effectiveness of POT, we collect and establish a new real-world dataset, named PATag, from Dianping. PATag contains 555, 297 reviews and 135, 586 opinion tags from 68, 732 users and 58, 643 products. Experiments conducted on PATag show that POT significantly outperforms state-of-the-art baselines in terms of generation and ranking metrics on personalized abstractive opinion tagging.

To sum up, our contributions in this paper are as follows:

- To the best of our knowledge, we are the first to generate personalized opinion tags in an abstractive fashion.
- We propose an end-to-end model, POT, to generate a ranked list of aspect and opinion tag pairs.
- POT is able to track user preferences by integrating implicit behavior with explicit feedback.
- Experimental results conducted on the PATag dataset verify the effectiveness of our proposed model. We show that POT significantly outperforms baselines.
- We collect a large-scale dataset named PATag, consisting of user historical reviews, product reviews, and opinion tags.

2 RELATED WORK

2.1 Keyphrase generation

A lot of research has been conducted on generating keyphrases to summarize important information from multi-source user-generated content in specific scenarios, such as social media platforms [32, 41, 45, 48] and e-commerce scenarios [16, 22, 23, 25, 46]. Keyphrase generation in social media concerns distilling salient information from large numbers of posts to quickly summarize current events. Zhang et al. [47] present a neural keyphrase extraction framework for microblog posts that takes their conversation context into account to alleviate the problem of data sparseness. Wang et al. [41] explore the abstractive approaches, and propose a topic-aware model to generate key phrases by modeling the potential

topic representations in tweets. In the e-commerce scenario, the keyphrase is usually expressed as tags, tips, or product titles. Sun et al. [34] propose a multi-source pointer network to generate informative and fluent short titles by copying words from not only the source title but also the background knowledge. Li et al. [24, 25] jointly model the tasks of user rating prediction and tip generation, to generate abstractive tips with good linguistic quality, simulating user experience and feelings. Yang et al. [44] design a query-aware tip generation framework, which explores user intent to help users gain quick insights into the search results.

Different from existing work that considers generating a short keyphrase in a single dimension, we consider a comprehensive summary from multiple aspects.

2.2 Opinion summarization

Opinion summarization has become an active research area in recent years. Early studies on opinion summarization focus on extracting salient sentences from the original review text [2, 7, 17, 27, 43]. Xiong and Litman [43] propose a novel unsupervised extractive approach for opinion summarization by exploiting review helpfulness ratings. Angelidis and Lapata [2] utilize weakly supervised learning methods to extract review summaries by combining the tasks of aspect extraction and sentiment prediction. Recently, abstractive approaches have also been explored [3, 11, 13, 14, 33, 38, 39]. Wang and Wan [38] propose a self-supervised framework to generate opinion summaries by exploiting the intra- and inter-group invariances of aspect and sentiment.

Abstractive review summarization are also receiving more and more attention: Li et al. [23] design an attribute-aware sequence network for personalized review summarization, which mainly considers word-using habits or writing styles of different users. Chan et al. [8] propose a novel dual-view model with inconsistency loss to jointly improve the performance of review summarization and sentiment classification. Nguyen et al. [30] generate review summaries for specific aspects of the product in an unsupervised way, reducing the generation of generic and uninformative content.

The above summarization approaches only consider textual information while ignoring the interaction between users and products. In contrast, we focus on personalized opinion tags generation by extracting user preference information from user explicit feedback and their implicit behavior w.r.t. the product.

3 PRELIMINARIES

Before detailing our method, we first provide important concepts and formulate the research problem. Then we introduce the hierarchical heterogeneous review graph applied in our model.

3.1 Problem formulation

Table 1 lists important notations used in this paper.

We first define the notions of aspects and opinion tags. We denote a set of aspects as $A = \{a_1, a_2, \ldots, a_n\}$, where each aspect $a_i \in A$ is defined as a manually created keyword that expresses characteristics of a product, e.g., taste, service, etc. (in a restaurant). We define an opinion tag $\tau = [w_1, w_2, \ldots, w_l]$ to be a sequence of l words on a specific aspect to describe a product. The task of personalized opinion tagging aims to automatically generate a ranked list of aspect-opinion tag pairs of a specific product for a specific user.

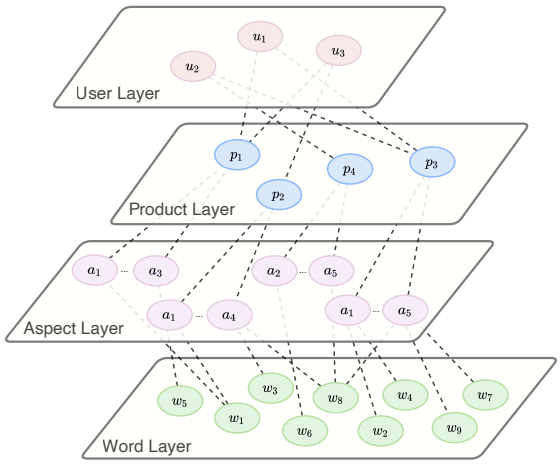

Figure 3: The architecture of HHRG. There are four hierarchical layers in HHRG: users, products, aspects, and words in reviews.

We suppose there are a set of users U and a set of products P in the task of personalized opinion tagging. Given a user $u \in U$ and a product $p \in P$, we denote the historical reviews of user u and product p as R_u and R_p, respectively. We define a set of products $P_u = \{p_1, p_2, \ldots, p_m\}$ as products that user u reviewed. Given a user u, a product p, historical reviews R_u and R_p, the objective of the task of personalized opinion tagging is to generate an ordered sequence of aspect-opinion tag pairs, i.e., $Y^{u,p} = [Y_1^{u,p}, Y_2^{u,p}, \ldots, Y_n^{u,p}]$, in which $Y_i^{u,p} = [a_i, \tau_i]$ contains an aspect $a_i \in A$, and an aspect-aware opinion tag τ_i that describes the aspect. Note that the opinion tag that appears in high-ranking pairs in $Y^{u,p}$ has a higher level of user preferences. In order to further explore user preferences through different types of behavior, the task of personalized opinion tagging takes as input the behavior information between user u and products that have been **c**licked, **f**avored, or **o**rdered by u, i.e., $P_{B,u} = P_{C,u} \cup P_{F,u} \cup P_{O,u}$.

3.2 Hierarchical heterogeneous review graph

We design a hierarchical heterogeneous review graph (HHRG) for tracking user preferences from reviews. Formally, a graph is denoted as $G = (V, E)$, where V and E are sets of nodes and edges in the graph. As shown in Fig. 3, a HHRG consists of four layers from bottom to top: a word layer, an aspect layer, a product layer, and a user layer. Correspondingly, four types of nodes exist in V: words W, aspects A, products P, and users U. Three types of edges comprise E: user-product edges E_{UP}, product-aspect edges E_{PA}, and aspect-word edges E_{AW}. An edge in E_{UP} reflects an explicit relation between a user and a product, i.e., whether the user has written reviews for the product. An edge in E_{PA} indicates an aspect involved in the historical reviews of the product, whereas an aspect-word edge in E_{AW} reflects a linkage between a word and the aspect to which the word belongs.

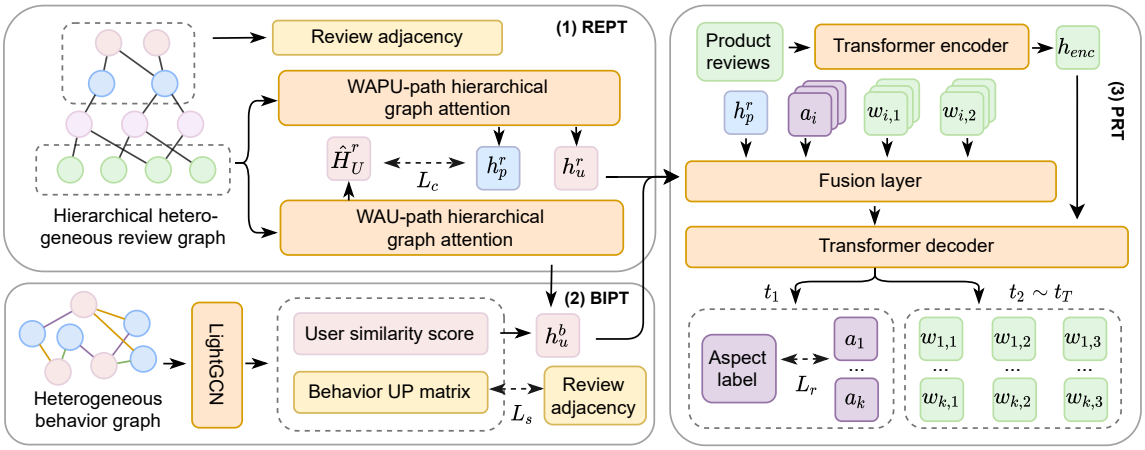

Figure 4: An overview of POT. POT is composed of three components: review-based explicit preference tracker (REPT), behavior-based implicit preference tracker (BIPT), and personalized rank-aware tagging (PRT). Note that nodes in red/blue/purple/green represents user/product/aspect/word, and edges in different colors in behavior graph mean different behavior.

Table 1: Glossary.

Symbol	Description
U	a set of users
P	a set of products
A	a set of aspects
W	a set of words
τ	an opinion tag
$Y_i^{u,p}$	an aspect-opinion tag pair of user u to product p in aspect a_i
R_u, R_p	historical reviews of user u and product p
P_u	a set of products reviewed by user u
$P_{B,u}$	a set of products that user u ever clicked, favored and ordered
$h_u^r, h_p^r, h_{a_i}^r, h_w^r$	the node representation of user u, product p, aspect a_i, and word w in HHRG based on the WAPU meta-path
$\hat{h}_u^r$	the node representation of user u in HHRG based on the WAU meta-path
S^c, S^f, S^o	the adjacent matrix of clicks, favor, and purchase behaviors
$\widetilde{H}_U^b, \widetilde{H}_P^b$	the node representation of users and products in HBG

4 METHOD

In this section, we detail our proposed method, named POT. As shown in Fig. 4, POT consists of a review-based explicit preference tracker (REPT), a behavior-based implicit preference tracker (BIPT), and a personalized rank-aware tagging (PRT).

4.1 Encoder and decoder

Based on an encoder-decoder framework, we use a transformer architecture [36] as the backbone of our framework. Before detailing each component, we first detail the encoder and decoder structure in POT.

Encoder. Given a product's historical reviews R_p, we first map each word $w \in R_p$ into a hidden representation by the word embedding layer and positional embedding layer:

$$h^{(0)} = \text{Embed}(w) + \text{Pos}(w). \tag{1}$$

where $h^{(0)}$ represents the initial hidden states. Then we stack L transformer layers to encode the global contextual information of reviews, i.e., $h_{enc} = h^{(L)}$. The hidden representation of layer l is computed as follows:

$$h^{(l)} = \text{FFN}(z^{(l)}), \tag{2}$$

$$z^{(l)} = \text{MultiHead}(W_q h^{(l-1)}, W_k h^{(l-1)}, W_v h^{(l-1)}), \tag{3}$$

where $\text{FFN}(\cdot)$ is a point-wise feed-forward layer with ReLU as the activation function; $\text{MultiHead}(\cdot)$ is the multi-head self-attention in transformers [36]; W_q, W_k, and W_v are trainable weights for subspace projection. Each transformer layer performs a normalization operation after residual connection.

Decoder. Given the input x, the decoder infers the target output y by maximizing the conditional probability $P(y|x)$. The decoder factorizes $P(y|x)$ into a product of conditional probabilities of the target output at each decoding step given the encoded hidden states h_{enc} and all the previous target output sequence:

$$P(y|x) = \prod_{t=0}^{T} P(y_t|y_1, y_2, \ldots, y_{t-1}, h_{enc}), \tag{4}$$

where T is the length of the target sequence and y_t denotes the t-th token of sequence y. Like the encoder, the decoder also stacks multiple transformer layers. Furthermore, each transformer layer of the decoder performs attention on the source-side context.

4.2 Review-based explicit preference tracker

The review-based explicit preference tracker (REPT) aims to obtain the review-based hidden representation of user preferences and product, i.e., h_u^r and h_p^r, based on the HHRG introduced in §3.2.

As illustrated in Fig. 3, HHRG consists of four layers: word layer, aspect layer, product layer and user layer. Representations of nodes in the graph are obtained from the representation of the word layer through the propagation and aggregation of heterogeneous nodes. There are two critical stages for representation learning of HHRG:

(1) node representation updates; and (2) information propagation across the hierarchical heterogeneous review graph.

4.2.1 Node representation updates. After the initialization of a node representation, we apply a graph attention network (GAT) [37] to update each node's representation based on its associated nodes. Formally, given a node i's representation h_i and its associated nodes N_i in the layer underneath, we update the representation of node i to get h'_i:

$$h'_i = \text{GAT}(h_i, \{h_j, j \in N_i\}). \tag{5}$$

where GAT computes the attention scores between the node and its neighborhoods based on a shared attention mechanism. As the heterogeneous characteristics of HHRG, we map all the nodes into the same vector space before aggregation. For each node $j \in N_i$, we compute the attention weight α_{ij} between node i and j as follows:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in N_i} \exp(e_{ik})}, \quad e_{ij} = \text{FFN}([h_i, Wh_j]), \tag{6}$$

where $\text{FFN}(\cdot)$ is a feedforward layer followed by a LeakyReLU activation; W is a trainable parameter in order to project the representation of the neighbor nodes into the representation space of node i. Then we compute the node representation as a weighted sum of its associated nodes representations, i.e., $h'_i = \sum_{j \in N_i} \alpha_{ij} h_j$. We employ multi-head attention to enhance the representation power. Hence we perform K independent attention mechanisms and get $h'^{(1)}_i, h'^{(2)}_i, \ldots, h'^{(K)}_i$. Then we aggregate these representations as follows:

$$h'_i = \text{Agg}(h'^{(1)}_i, h'^{(2)}_i, \ldots, h'^{(K)}_i), \tag{7}$$

where *Agg* indicates a concatenation operation or an averaging operation. We stack L layers of GAT for updating representations. Specifically, we perform the averaging aggregation operation in the final layer and the concatenation operation in other layers.

4.2.2 Meta-path based propagation. To track the reviewed-based hidden representation of user preferences, we propose a word-aspect-product-user (WAPU) meta-path to propagate preferences from word-level to user-level on HHRG. A WAPU meta-path propagates information from words of a product's historical reviews to products, and then to the user. We add an aspect layer between the product layer and the word layer for preference filtering and aggregation. Given a node for user u and its associated nodes N_u in the product layer, the information aggregation process is calculated as follows:

$$h^r_u = \text{GAT}(h^{r\,(0)}_u, \{h^r_p, p \in N_u\}), \tag{8}$$

where h^r_u is the review-based user preference and $h^{r\,(0)}_u$ is the initial representation. Here, h^r_p is the final representation of the node for each product $p \in N_u$, which is calculated as follows:

$$h^r_p = \text{Con}(h^r_{a_1}, h^r_{a_2}, \ldots, h^r_{a_n}), \tag{9}$$

where $\{a_1, a_2, \ldots, a_n\}$ are the associated nodes of p in the aspect layer, whereas $\text{Con}(\cdot)$ is the concatenate operation. For each node $a_i \in \{a_1, a_2, \ldots, a_n\}$, we calculate $h^r_{a_i}$ using representations of its associated nodes N_{a_i} in the word layer:

$$h^r_{a_i} = \text{GAT}(h^{r\,(0)}_{a_i}, \{h^r_w, w \in N_{a_i}\}). \tag{10}$$

where h^r_w denotes the word representation in the word layer. Note that all $\text{GAT}(\cdot)$ operations in meta-path propagation will stack multiple layers for better aggregation. By exploring interactions between users and products, our meta-path based propagation enriches user/product representations and alleviates the sparseness problem in reviews.

4.2.3 Contrastive learning. Although we obtain representations using the meta-path based propagation, it is still noisy to track user preferences from a large set of reviews due to the long propagation path. To this end, we use shorter meta-paths as supervision signals and adopt a contrastive learning model to denoise. The model aggregates the representations of users with similar preferences and vice versa. We assume that products and users who have reviewed products should have similar preference representations.

For a product p, we denote a user who provided positive reviews as a positive sample u^+; whereas we randomly choose a user who has no interaction with p to form a negative sample u^-. The objective of contrastive learning is to minimize the following loss:

$$\mathcal{L}_c = -\log \frac{\exp\left(h^r_p \cdot \hat{h}^r_{u^+}\right)}{\sum_{u^* \in \{u^+, u^-\}} \exp\left(h^r_p \cdot \hat{h}^r_{u^*}\right)}, \tag{11}$$

where h^r_p, $\hat{h}^r_{u^+}$ and $\hat{h}^r_{u^-}$ denote representations of product p, positive user u^+ and negative user u^-. By maximizing $(h^r_p \cdot \hat{h}^r_{u^+})$, the contrastive loss forces their semantic representations to project close to each other.

A user representation propagated by a WAPU meta-path may suffer from error accumulation of product representations. Therefore, we obtain a user representation in contrastive learning directly from the word layer and aspect layer. We build a word-aspect-user (WAU) meta-path with three layers of nodes using the user's historical reviews. The representation of the user is obtained in a similar manner as the calculation of the representation of a product in Eq. 9, which is denoted as $\hat{h}^r_u$.

4.3 Behavior-based implicit preference tracker

The behavior-based implicit preference tracker (BIPT) aims to infer user preferences from implicit feedback based on user behavior. We first build a heterogeneous behavior graph (HBG) to maintain the behavior between the users and products. Then we obtain a representation of the users using LightGCN [15]. And finally, we aggregate the representations of similar users as the final representation of implicit user preferences.

4.3.1 Behavior graph building. Formally, the behavior graph between the users and products is represented by an adjacent matrix S, where the entry $S(u, p) = n$ indicates that u has n interactions with p. We build three directed behavior graphs for three types of behavior in HBG, i.e., clicks S^c, favors S^f, and orders S^o. Note that the matrix $S^* \in \mathbb{R}^{|U| \times |P|}$, where $|U|$ and $|P|$ denote the number of users the products in the behavior graph, respectively.

4.3.2 Implicit preference tracking. Taking clicking behavior as an example, after the initialization of product and user embeddings, LightGCN is applied to obtain the representation of each node by aggregating the information from its neighbors. To capture the semantics with different granularities, we obtain the final node representation by combining the representations of each layer in LightGCN. The representations of all user nodes is calculated as

follows:

$$H_U^c = \frac{1}{L+1}\sum_{l=0}^{L} E_P^{(l)}, \tag{12}$$

where L is the number of stacked LightGCN layers; $E_P^{(l)}$ is the representations of all product nodes after l LightGCN layers and it is calculated by:

$$E_P^{(l)} = D^{-1}S^c E_P^{(l-1)}, \tag{13}$$

where S^c is the adjacency matrix of the clicking graph; D is the diagonal degree matrix of S^c and $E_P^{(0)}$ represents the embeddings of products. By aggregating the user representation obtained from each behavior graph, we compute the representation of user in HBG:

$$\widetilde{H}_U^b = W^T([H_U^c, H_U^f, H_U^o]), \tag{14}$$

where $W \in \mathbb{R}^{3\times d,d}$ is a trainable parameter for semantic space projection, and d is the hidden size of user representation.

As illustrated before, since there is no semantic information in the propagation, we only use it to calculate the similarity between two users. Then we obtain the implicit preference of users by aggregating the review-based preference of other users based on the similarity.

For each user u, we calculate a similarity score between u and all other users by conducting an inner product of their behavior representations in $\widetilde{H}_U^b$. Then, we select neighbors with top-k_s highest scores as the user most similar neighbors. Finally, we aggregate the review-based representations of similar neighbors to obtain user u's implicit preference h_u^b. So we have:

$$\begin{aligned} h_u^b &= \sum_{u_i \in N_u} w_{u,u_i}\hat{h}_{u_i}^r, \\ w_{u,u_i} &= \frac{\exp\left((\widetilde{h}_u^b)^T \cdot \widetilde{h}_{u_i}^b\right)}{\sum_{u_j \in N_u}\exp\left((\widetilde{h}_u^b)^T \cdot \widetilde{h}_{u_j}^b\right)}, \\ N_u &= \mathrm{TopK}(\widetilde{h}_u^b \cdot (\widetilde{H}_U^b)^T), \end{aligned} \tag{15}$$

where $\hat{h}_{u_i}^r$ is the user preference obtained from the meta-path WAU in §4.2.3; $\widetilde{h}_u^b$ and $\widetilde{H}_U^b$ is the user representation obtained by Eq. 14, where $\widetilde{H}_U^b$ is representation of all users; w_{u,u_i} is the calculated similarity between user u and his neighbor u_i.

4.3.3 Neighbor denoising. As user implicit behaviors usually are noisy [42], we perform relation denoising to select the most similar users on the behavior graph. These identified similar neighbors are the nodes that contribute most to personalized opinion tagging. To make the relations between the users and products predicted by the model more reliable, we reconstruct the user's explicit behaviors, i.e., reviews, on the behavior graph.

To impose the above constraint, we append a fully-connected multi-layer perception (MLP) after LightGCN layers for relation prediction. Here, the LightGCN and MLP layer acts as a concrete autoencoder [1], with the former as encoder, and then the latter as the decoder. After encoding by LightGCN, we get the representation of all users $\widetilde{H}_U^b$ and the representation of all products $\widetilde{H}_P^b$ according to Eq. 14. By forwarding an MLP, we obtain the scores of the relation between each user and product:

$$I^b = \mathrm{Sigmoid}(\widetilde{H}_U^b W \widetilde{H}_P^b), \tag{16}$$

where $W \in \mathbb{R}^{d\times d}$ is a training parameter and I^b is the final score matrix where each entry represents the score of the relation between a specific user and product. Also, the review interaction between user and product is constructed as a matrix I^r, where $I^r(u,p)$ equals 1 when the user u has written positive reviews for product p, and 0 for other situations. To make the relations predicted by the model reliable, we force the model to predict I^b as similar to I^r as possible. The select loss is expressed as:

$$\mathcal{L}_s = -\frac{1}{|I^r|}\sum I^r \log I^b + (1-I^r)\log(1-I^b), \tag{17}$$

where $|I^r|$ is the number of entries in I^r.

4.4 Personalized rank-aware tagging

As illustrated above, we extract the explicit and implicit user preferences based on REPT and BIPT. In this section, we fuse these preferences to generate personalized opinion tags. PRT first predicts the scores of each candidate aspect, and then selects those aspects with the top-k_g highest scores as the indicators to generate opinion tags. Finally, PRT predicts an opinion tag for each selected aspect and merges them as a sequence following the order of aspects.

At each decoding step t, we combine the input x_t and personalized representations within a Fusion layer. Then we feed the result into a TransDec decoder:

$$\begin{aligned} e_t &= \mathrm{Fusion}([h_u^r, h_u^b, x_t]), \\ p(y_t) &= \mathrm{TransDec}([e_1, e_2, \ldots, e_t]), \end{aligned} \tag{18}$$

where h_u^r, h_u^b are the representation of user u in HHRG and HBG; Fusion is implemented as a linear layer; TransDec works as a normal transformer decoder, containing three components: multi-head self-attention, multi-head cross-attention, and FFN, each of which performs layer normalization after residual connections. Note that multi-head cross-attention is integrated into the basic information of the product, that is, taking decoder hidden state at t as its query and the hidden representation in the final layer of a transformer encoder h_{enc} as its key and value.

4.4.1 Rank-aware aspect generation. At t_1, we set x_1 as the representation of product p in HHRG, i.e., $x_1 = h_p^r$. By using Eq. 18, we use x_1 to decode the scores of each aspect $a_i \in A$, i.e., $y_1 = a_i$. Thus we obtain the score $p(a_i)$ for each aspect a_i. To represent the preference of user u, we use a rank-based loss function to supervise the generation of aspects.

Given a set of aspects A, we use g_i and l_i to represent the score of the i-th aspect in the generated aspect sequence and label sequence. As a widely used pairwise loss function, the logistic loss based on cross entropy [40] is adopted here:

$$L(g,l) = \sum_{i=1}^{n}\sum_{j=1}^{n}\log_2(1+e^{-\sigma(g_i-g_j)}), l_i > l_j, \tag{19}$$

where σ is a hyper-parameter. Used in RankNet [4, 5], the intuition of this loss is to apply a penalty on the out-of-order pair (i,j) that has $l_i > l_j$ but $g_i < g_j$. Inspired by LambdaRank [6], we optimize Eq. 19 by adapting a logistic loss and reweight each user-product pair using ΔNDCG in each iteration:

$$\mathcal{L}_r = \sum_{l_i > l_j}\Delta\mathrm{NDCG}(i,j)\log_2(1+e^{(g_j-g_i)}), \tag{20}$$

where ΔNDCG is defined as the absolute difference between the NDCG values when two aspects i and j are swapped.

4.4.2 Opinion tag generation. We select the aspects with top-k_g highest scores and denote them as $(a_1, a_2, \ldots, a_{k_g})$. Then the transformer decoder takes each selected aspect as the first token of a sequence to obtain the corresponding personalized opinion tags:

$$x_t = \begin{cases} \text{Emb}(a_i), & \text{if } t = 2, \\ \text{Emb}(y_{i,t-1}), & \text{if } t > 2, \end{cases} \qquad (21)$$
$$y_{i,t} = w_{i,t-1}, t > 1,$$

where $w_{i,t-1}$ denotes the $(t\text{-}1)$-th token in the opinion tag sequence for aspect a_i. To maximize the prediction probability of the target opinion tags, we use the negative log-likelihood loss for supervision:

$$\mathcal{L}_g = -\sum_{i=1}^{k_g}\sum_{t=1}^{T} \log p(y^*_{i,t}|y^*_{i,1}, \ldots, y^*_{i,t-1}), \qquad (22)$$

where $p(y^*_{i,j}|y^*_{i,1}, \ldots, y^*_{i,j-1})$ is the generation probability of the j-th target opinion tag token for aspect a_i and T is the maximum number of decoding steps.

We adopt a multi-task learning framework to jointly minimize the contrastive loss, select loss, rank loss, and generation loss. The objective function is:

$$\mathcal{L} = \lambda_1\mathcal{L}_c + \lambda_2\mathcal{L}_s + \lambda_3\mathcal{L}_r + \lambda_4\mathcal{L}_g, \qquad (23)$$

where λ_1, λ_2, λ_3, λ_4 are hyper-parameters that control the weights of these four losses. Thus our model is trained in an end-to-end way.

5 EXPERIMENTAL SETUP

5.1 Research questions

Our research questions are: (RQ1) How does the proposed model POT perform compared with the baselines? (See §6.1.) (RQ2) How do different components of POT contribute to the overall performance? (See §6.2.) (RQ3) Are the generated aspects and tags consistent? (See §6.3.) (RQ4) What is the effect of the number of similar user neighbors k_s in §4.3.3? (See §6.4.) (RQ5) Is POT able to generate personalized tags for a specific user? (See §6.5.)

5.2 Dataset

Due to a lack of public datasets for personalized opinion tagging, we propose a **P**ersonalized **A**spect-aware Opinion **Tag**ging dataset (**PATag**) by crawling reviews from the Dianping portal. Dianping is a leading Chinese e-commerce platform, where customers can write reviews for products such as restaurants, hotels, etc. We focus on the domain of gastronomy, which received the most attention from users in Dianping. We collect user reviews and implicit behavior from January 1st to July 1st, 2021 in Beijing and Shanghai. In total, our dataset involves more than 550k reviews from 68k users and 58k products.

Our dataset contains annotations of opinion tags for each sentence by manually labeling how the user feels about a certain aspect of the product. In PATag, we also label the semantic correlation between the opinion tags and aspects. For each user, opinion tags in the latest reviews are selected as the ground-truth, and their historical reviews are used as input to explore preferences. As multiple opinion tags may link to the same aspect in reviews, we select the one with the highest confidence.[3] For each product, we sort opinion tags according to the distribution of aspects in historical reviews. In order to ensure data quality, we remove users with less than three reviews to filter out extremely sparse nodes and remove users who are suspected of being or using ghostwriters. Given products reviewed by a user, we select products that have positive after-sales ratings as the set of products favored by the user. At the same time, we only select reviews covering more than four aspects for prediction. Finally, we obtain 135, 586 samples for our task, where each sample consists of a ranked list of aspect-opinion tag pairs. For text pre-processing, we tokenize texts using the Jieba toolkit[4] and remove words with low-frequency. We finally get a vocabulary of size 44k. Besides, we also collect behavioral information during the same time period, which includes 15 million actions including clicking, favoring, and ordering. The statistics of our dataset are shown in Table 2.

Table 2: Dataset description. "ATN" and "ATL" indicate the average number of tags for a sample and the average length of each tag, respectively.

Review		Behavior		Samples	
User	68,732	Click	12,093,919	Number	135,586
Product	58,642	Favor	2,382,110	ATN	3.4
Review	555,297	Order	1,323,805	ATL	8.3

5.3 Baselines and comparisons

In the context of RQ1, we set up experiments to compare POT against several baselines. We list the baselines that we consider in Table 3 and more details are as follows:

(1) **TextRank** is an unsupervised algorithm based on weighted graphs; it is widely used to extract and rank key phrases from source sentences [28]. (2) **RNN** is a classic seq2seq model with attention implemented by a bi-directional GRU layer [10]. (3) **Trans** is a traditional transformer framework that has been widely used in abstractive generation [36]. (4) **KOBE** is used to generate personalized product descriptions in e-commerce based on user attributes, product titles and a knowledge graph [9]. We remove the knowledge graph part and select the most frequent aspect in a user's historical reviews as the user attributes. (5) **Trans_Q** is a query-aware tip generation framework that extracts user intent with query and integrates it into generation [44]. We replace the query with the most frequent aspect in a user's historical reviews. (6) **AOT** is a state-of-the-art model for abstractive opinion tagging task that aims to generate a ranked list of opinion tags from a large number of reviews [26]. Its rank only represents the collective interests of users and does not reflect personalized information.

Since most of the baselines do not involve ranking tasks, the generated results of the baselines are post-processed to measure the ranking performance. For KOBE and Trans_Q, we first generate the aspect and then generate the tag. If the generated aspect is consistent with the aspect in the label, it is regarded as a match. For other baselines, since there is no explicit aspect involved in our input, we will adopt the approach of Li et al. [26]. If the generated

[3]Confidence represents the probability that an opinion tag belongs to an aspect.
[4]https://github.com/fxsjy/jieba

Table 3: Our method and baselines used for comparison.

Acronym	Gloss	Reference
TextRank	Extract and rank key phrases for summarization	[28]
RNN	Classic seq2seq model with bi-directional GRU for summarization	[10]
Trans	Traditional transformer framework for summarization	[36]
KOBE	Knowledge-based personalized product description	[9]
Trans_Q	Query-aware tip generation	[44]
AOT	Traditional abstractive opinion taggging	[26]
POT	Personalized abstractive opinion taggging	This paper

tag and any tag in the label have two or more words in common, it is regarded as a match.

Furthermore, to address RQ2, we conduct ablation studies to better understand the effectiveness of two core mechanisms: (i) the REPT module mines user preferences from user-reviewed products, abandoning the traditional method based on user history reviews; (ii) BIPT module exists to demonstrate that implicit behaviors can further enhance user preferences. We evaluate variants of our model by removing each of these component and then assessing the performance on the PATag dataset.

5.4 Evaluation metrics

We conduct both automatic and human evaluations.

Automatic evaluation. We adopt the following metrics to evaluate ranking and generation performance: (i) **Ranking metrics.** We employ two information retrieval metrics to evaluate the personalization of opinion tags in terms of aspect ranking: the *F1@3, 5* score fuses the precision score and the recall score of the generated aspect; and the normalized discounted cumulative gain (*NDCG@3, 5*) score is used to measure the ranking performance. (ii) **Generation metrics.** As generating long sentences would defeat our purpose, we employ classical word-overlap based metrics *BLEU-1,2* [31] to assess whether the generated opinion tags are concise and informative. Besides, we adopt the *Distinct-1,2* score [21] to measure the sample-level diversity of the generated opinion tags. Additionally, following [9] we use *Lexical Diversity-3, 5* to measure an overall diversity of all the generated opinion tags.

Human evaluation. We randomly sample 500 generated opinion tags by our model and the baselines, as well as ground-truth results. We recruit five professional annotators to evaluate the opinion tags generated by different models in terms of the following three metrics: *Tag Sequence Fluency* measures if the generated opinion tags are smooth; *Aspect Correctness* evaluates whether the generated aspects are correct; and *Aspect Diversity* measures the diversity of aspects. Five annotators are asked to rate each generated sequence with a score ranging from 1 (bad) to 5 (excellent). Model names were masked out during evaluation.

5.5 Implementation details

We conduct our experiments with the batch size set to 16 and use the validation loss for early stopping. We use the Adam optimizer [20] with a learning rate of $5e^{-5}$. We use Tencent AI Lab Chinese Embeddings[5] for word embedding initialization with size of 200. The rest of the parameters are randomly initialized. The representation size of hierarchical heterogeneous graph nodes, including aspect, product and user, are all set to 256, while the product and user representation sizes are 64 in the behavior graph. All transformer-based models have 256 hidden units. For all stackable base layers, we set up $L = 2$, and the number of heads for multi-head attention is set to 2. We use $k_s = 15$ in §4.3.3, and to ensure that sufficient comprehensive information is generated, we set $k_g = 6$ in §4.4.2. For the optimization objective, we let the weight parameters $\lambda_1 = \lambda_2 = \lambda_4 = 1$ and $\lambda_3 = 0.75$. All hyperparameters and models are selected on the validation set and the results are reported on the test set.

6 EXPERIMENTAL RESULTS

6.1 Overall performance

To answer RQ1, we conduct a comprehensive comparison between POT and the baselines. Based on the automatic evaluation results reported in Table 4, we find the following observations.

First, POT significantly outperforms all the baselines in terms of most evaluation metrics. For ranking evaluation, POT achieves an increase of 12.35%, 10.45%, 46.64%, and 46.00% over the best baseline in terms of F1@3, F1@5, NDCG@3, and NDCG@5, respectively. This is because POT tracks user preferences jointly from the user explicit reviews and implicit behavior, whereas other baselines either model user preferences purely from user attributes or ignore personalization. For generation evaluation, we find that POT offers the best performance over all baselines. In particular, POT achieves a 8.46% (2.15%) increase over Trans in terms of BLEU-1 (BLEU-2). The reason is that the decoder in POT takes accurate hidden representations of user preferences and generates accurate aspect-aware opinion tags to avoid dull generation.

Second, POT exhibits the best performance in terms of ranking metrics in the experiments. In terms of NDCG@3 (NDCG@5), POT offers an increase over Trans_Q and KOBE of up to 46.64% (50.00%) and 46.00% (54.04%), respectively. This is because POT tracks user preferences by modeling the related reviews with a hierarchical heterogeneous graph rather than plain text. KOBE and Trans_Q outperform other baselines, they extract user preferences from attributes while others ignore user preferences modeling altogether.

Third, POT can generate more personalized and diverse opinion tags in contrast with other baselines. POT achieves an increase of 2.25%, 4.79%, 48.67% and 48.99% over Trans in terms of Distinct-1, Distinct-2, LD-3, and LD-5, which shows the effect of PRT module in POT. Also, POT achieves an increase of 168% (218%) over KOBE in terms of LD-3 (LD-5), and offers comparable performance over KOBE in terms of Distinct-1 and Distinct-2. A potential reason is that KOBE generates descriptions by capturing the attributes of user preferences, which is able to provide sample-level diverse results. However, attributes of different users still have similar tokens, which decrease the diversity of the generated results. In

[5] https://ai.tencent.com/ailab/nlp/en/embedding.html

Table 4: The performance of different models on PATag. Boldface scores indicate best results, significant improvements over the best baseline are marked with * (t-test, $p < 0.05$).

Model	Ranking metrics				Generation metrics					
	F1@3	F1@5	NDCG@3	NDCG@5	BLEU-1	BLEU-2	Distinct-1	Distinct-2	LD-3	LD-5
TextRank	12.63	12.66	18.43	19.10	6.70	2.09	-	-	-	-
RNN	41.80	48.41	36.90	41.33	24.32	9.80	94.35	88.70	2.44	6.84
Trans	41.16	47.74	35.46	40.21	25.65	10.69	94.53	89.06	2.65	7.94
KOBE	46.93	46.00	50.07	50.66	24.51	10.31	**96.87**	**93.74**	1.47	3.71
Trans_Q	43.72	47.86	51.22	53.45	24.32	10.27	94.62	89.45	1.25	3.52
AOT	42.46	49.15	37.63	42.01	24.89	10.23	94.37	88.75	0.64	1.72
POT	**52.73***	**54.29***	**75.11***	**78.04***	**27.82***	**10.92**	**96.66**	**93.33**	**3.94***	**11.83***

Table 5: Human evaluation on PATag (with 25% annotations).

Model	Fluency	Correctness	Diversity
KOBE	3.97	2.05	3.76
Trans_Q	3.78	2.41	3.44
AOT	3.66	1.75	2.97
POT	**4.21**	**3.29**	**4.34**

Table 6: A comparison of different variations by masking out personalized module.

Model	F1@3	F1@5	NDCG@3	NDCG@5
POT	**52.73**	**54.29**	**75.11**	**78.04**
- REPT	51.52	53.00	74.67	76.99
- BIPT	52.07	54.18	73.13	76.82

contrast, POT can capture personalized information for each user by richer interactions in HHRG, which results in more diverse tokens for the same aspect for different users.

We also perform human evaluation on PATag for comparison in Table 5. We compute the average pairwise Cohen's kappa κ to measure the consistency between annotators, and find that $0.4 \leq \kappa \leq 0.6$ for all metrics. We find POT outperforms all baselines in terms of all metrics. This is due to accurate user preference tracking by the REPT and BIPT modules, leading to large improvements in terms of Correctness. In terms of Diversity, the non-personalized model, i.e., AOT, performs worse than other models with personalized modules, indicating that AOT only captures popular aspect preferences, while others capture more diverse features for different users. This result is consistent with our automatic evaluation results, confirming the effectiveness of our proposed POT model for tracking user preference.

6.2 Ablation study

Next, we turn to RQ2. As shown in Table 6, all components in POT contribute to its performance. We measure the performance of two personalized modules in aspect ranking. The performance of POT w/o REPT drops by 2.43% on F1@5 and 1.36% on NDCG@5, which indicates that our proposed method for mining user preferences based on user-reviewed products indeed resists sparseness of reviews. At the same time, we can observe that the performance of POT w/o BIPT drops by 1.58% in terms of NDCG@5, which indicates that implicit behaviors can further enhance the learning of personalized representation information. Besides, we see POT offers a minor increase over POT w/o BIPT up to 0.20% in terms of F1@5. This shows the noise is difficult to be completely eliminated in our dataset.

6.3 Consistency of aspects and tags

Turning to RQ3, we examine the semantic consistency of an aspect and its corresponding opinion tag given the generated aspect-opinion tag pairs. We select 3 most popular aspects from PATag. For each aspect, we randomly select 13 corresponding words from the generated opinion tags of POT, and plot them into a 2D plane using the t-SNE algorithm [35]. As shown in Fig. 5, we observe that the words always get close to the aspect to which the corresponding opinion tag belongs, and get far from other aspects. Besides, we see some words are far from all aspects, such as "pretty" and "nice," which are generated in multiple aspects.

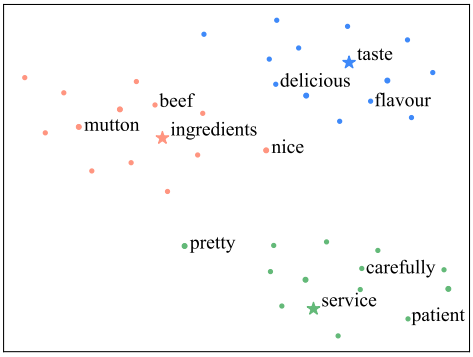

Figure 5: Visualization of fine-tuned embeddings of aspects (star) and opinion tags (point). Different aspects use different colors. We mark aspects and some corresponding words to their right.

6.4 Number of user similar neighbors

To address RQ4, we examine the performance of POT with different values of k_s (see §4.3.3), setting it to 10, 15, and 20, respectively. As shown in Fig. 7, compared to $k_s = 15$, $k_s = 10$ slightly decreases the model ranking and generation performance. This suggests that relatively small numbers of neighbors are not sufficient to enrich the user preference. When $k_s = 20$, the generation performance of POT significantly decreases although it is comparable to $k_s = 15$ on the ranking metrics, suggesting that more similar neighbors bring more

Input	The	taste	is	quite	authentic	...	Easy	to	be	full	...	Service	is	good	...
	With	a	mouthful	of	fresh	fragrance	...	It's	a	very	nice	environment	...	The	waiter
	is	patient	and	careful	...	The	restaurant	is	clean	and	tidy	...	Feel	good	...
AOT (A)	Overall it's quite good.					The taste is also good.				Service is nice.					
POT (A)	[Taste] Just melt in my mouth.					[Feeling] Satisfied for this eating.				[Environment] The environment is tidy and bright.					
Label (A)	[Taste] Sauce in the mouth.					[Feeling] Give the restaurant thumbs up!				[Environment] The desktop is clean.					
AOT (B)	Overall it's quite good.					The taste is also good.				Service is nice.					
POT (B)	[Feeling] Overall it's still pretty good.					[Taste] Foie gras sushi is delicious.				[Service] The meat is tender and tastes good.					
Label (B)	[Feeling] I'm stuffed.					[Service] Satisfied with the service.				[Taste] The meat is tender and tastes good.					

Figure 6: Case study for POT. The top is the product reviews, where words in blue/red/yellow obtain most attention for the tag generation of Taste/Feeling/Environment aspect for POT (A). Note that the deeper the color, more attention the word gets. Below are two generation examples for different users (A and B) for this product.

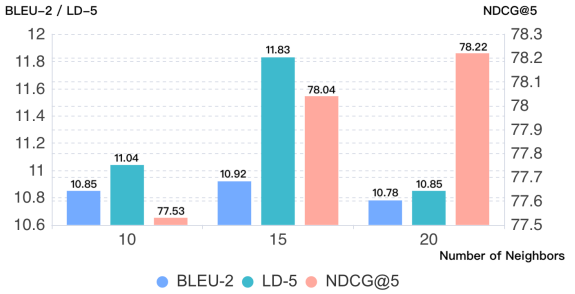

Figure 7: Performance when selecting different numbers of similar neighbors k_s for a user.

noise. POT achieves the best performance in terms of all metrics when $k_s = 15$. Hence, we infer that $k_s = 15$ is a trade-off between focusing on relevant information and removing irrelevant noise.

6.5 Case study

Finally, we address RQ5 with case studies to look into how our proposed model influences the generation process. Fig. 6 shows examples of opinion tags generated by AOT and POT for different users of the same product. It is clear that POT can focus on words of specific aspect when generating opinion tag for this aspect. Moreover, our model effectively captures the aspect preferences of different users and accurately generates the corresponding opinion tags. For the same product, our model generates an "environment" aspect for user A and a "service" aspect for user B, which is consistent with the user preference in the ground truth. For aspects that both users are concerned about, POT generates different tags for different users by capturing the writing style in the user historical reviews. Note that AOT only generates a list of opinion tags without personalization. We observe that AOT cannot generate diverse opinion tags for different users of the same product.

7 CONCLUSION

In this paper, we have focused on the task of personalized abstractive opinion tagging. To generate a ranked list of aspect-opinion tag pairs, we have tracked user preferences from user reviews and behaviors. To tackle the challenge about review sparsity and difficulty of integrating multiple signals, we have proposed an end-to-end model, namely POT. POT mainly consists of three components: *review-based explicit preference tracker* (RPET), *behavior-based implicit preference tracker* (BIET), and *personalized rank-aware tagging* (PRT). Extensive experiments conducted on a real-world dataset have shown the effectiveness of the proposed model.

Although we focused mostly on e-commerce portals, the proposed model is also broadly applicable to other scenarios with opinionated content, such as social media. However, a limitation of POT is that it still neglects product entities that play a key role in opinion tagging. As to future work, we aim to integrate external knowledge into our model to construct entity connections in reviews. In addition, mining fine-grained user preferences should provide more insights for personalized opinion tagging. As our work is based on real-world scenarios, we will deploy POT in a production environment to test its online performance in future.

REPRODUCIBILITY

To facilitate reproducibility of our results, the code and data used are available at https://github.com/MengxueZhao/POT.

ACKNOWLEDGMENTS

This work was supported by the National Key R&D Program of China with grant No. 2020YFB1406704, the Natural Science Foundation of China (61902219, 61972234, 62072279, 62102234), Meituan, the Hybrid Intelligence Center, a 10-year program funded by the Dutch Ministry of Education, Culture and Science through the Netherlands Organisation for Scientific Research, https://hybrid-intelligence-centre.nl. All content represents the opinion of the authors, which is not necessarily shared or endorsed by their respective employers and/or sponsors.

REFERENCES

[1] Abubakar Abid, Muhammad Fatih Balin, and James Zou. 2019. Concrete autoencoders for differentiable feature selection and reconstruction. *arXiv preprint arXiv:1901.09346* (2019).
[2] Stefanos Angelidis and Mirella Lapata. 2018. Summarizing Opinions: Aspect Extraction Meets Sentiment Prediction and They Are Both Weakly Supervised. In *Proceedings of EMNLP*. 3675–3686.
[3] Arthur Bražinskas, Mirella Lapata, and Ivan Titov. 2020. Few-Shot Learning for Opinion Summarization. In *Proceedings of EMNLP*. 4119–4135.
[4] Christopher Burges, Robert Ragno, and Quoc Le. 2006. Learning to rank with nonsmooth cost functions. *Advances in neural information processing systems* 19 (2006), 193–200.
[5] Chris Burges, Tal Shaked, Erin Renshaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and Greg Hullender. 2005. Learning to rank using gradient descent. In *Proceedings of ICML*. 89–96.

[6] Christopher JC Burges. 2010. From RankNet to LambdaRank to LambdaMART: An overview. *Learning* 11, 23-581 (2010), 81.

[7] Giuseppe Carenini, Jackie Chi Kit Cheung, and Adam Pauls. 2013. Multi-document summarization of evaluative text. *Computational Intelligence* 29, 4 (2013), 545–576.

[8] Hou Pong Chan, Wang Chen, and Irwin King. 2020. A unified dual-view model for review summarization and sentiment classification with inconsistency loss. In *Proceedings of SIGIR*. 1191–1200.

[9] Qibin Chen, Junyang Lin, Yichang Zhang, Hongxia Yang, Jingren Zhou, and Jie Tang. 2019. Towards knowledge-based personalized product description generation in e-commerce. In *Proceedings of KDD*. 3040–3050.

[10] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).

[11] Giuseppe Di Fabbrizio, Amanda Stent, and Robert Gaizauskas. 2014. A hybrid approach to multi-document summarization of opinions in reviews. In *Proceedings of INLG*. 54–63.

[12] Shuai Ding, Yeqing Li, Desheng Wu, Youtao Zhang, and Shanlin Yang. 2018. Time-aware cloud service recommendation using similarity-enhanced collaborative filtering and ARIMA model. *Decision Support Systems* 107 (2018), 103–115.

[13] Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: A graph based approach to abstractive summarization of highly redundant opinions. (2010).

[14] Shima Gerani, Yashar Mehdad, Giuseppe Carenini, Raymond Ng, and Bita Nejat. 2014. Abstractive summarization of product reviews using discourse structure. In *Proceedings of EMNLP*. 1602–1613.

[15] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proceedings of SIGIR*. 639–648.

[16] Sharon Hirsch, Slava Novgorodov, Ido Guy, and Alexander Nus. 2021. Generating tips from product reviews. In *Proceedings of WSDM*. 310–318.

[17] Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of KDD*. 168–177.

[18] Parisa Kaghazgaran, Jianling Wang, Ruihong Huang, and James Caverlee. 2020. Adore: Aspect dependent online review labeling for review generation. In *Proceedings of SIGIR*. 1021–1030.

[19] Minjae Kim, SungHwan Jeon, Heeseong Shin, Wonseok Choi, Haejin Chung, and Yunmook Nah. 2019. Movie recommendation based on user similarity of consumption pattern change. In *Proceedings of AIKE*. 317–319.

[20] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).

[21] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055* (2015).

[22] Junjie Li, Haoran Li, and Chengqing Zong. 2019. Towards personalized review summarization via user-aware sequence network. In *AAAI*. 6690–6697.

[23] Junjie Li, Xuepeng Wang, Dawei Yin, and Chengqing Zong. 2019. Attribute-aware sequence network for review summarization. In *Proceedings of EMNLP-IJCNLP*. 3000–3010.

[24] Piji Li, Zihao Wang, Lidong Bing, and Wai Lam. 2019. Persona-aware tips generation.. In *Proceedings of WWW*. 1006–1016.

[25] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. Neural rating regression with abstractive tips generation for recommendation. In *Proceedings of SIGIR*. 345–354.

[26] Qintong Li, Piji Li, Xinyi Li, Zhaochun Ren, Zhumin Chen, and Maarten de Rijke. 2021. Abstractive opinion tagging. In *Proceedings of WSDM*. 337–345.

[27] Yue Lu, ChengXiang Zhai, and Neel Sundaresan. 2009. Rated aspect summarization of short comments. In *Proceedings of WWW*. 131–140.

[28] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into text. In *Proceedings of EMNLP*. 404–411.

[29] Luong Vuong Nguyen, Min-Sung Hong, Jason J Jung, and Bong-Soo Sohn. 2020. Cognitive similarity-based collaborative filtering recommendation system. *Applied Sciences* 10, 12 (2020), 4183.

[30] Thi Nhat Anh Nguyen, Mingwei Shen, and Karen Hovsepian. 2021. Unsupervised Class-Specific Abstractive Summarization of Customer Reviews. In *Proceedings of ECNLP*. 88–100.

[31] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL*. 311–318.

[32] Jishnu Ray Chowdhury, Cornelia Caragea, and Doina Caragea. 2019. Keyphrase extraction from disaster-related tweets. In *Proceedings of WWW*. 1555–1566.

[33] Xuemeng Song, Liqiang Jing, Dengtian Lin, Zhongzhou Zhao, Haiqing Chen Chen, and Liqiang Nie. 2022. V2P: Vision-to-Prompt based Multi-Modal Product Summary Generation. In *Proceedings of SIGIR*.

[34] Fei Sun, Peng Jiang, Hanxiao Sun, Changhua Pei, Wenwu Ou, and Xiaobo Wang. 2018. Multi-source pointer network for product title summarization. In *Proceedings of CIKM*. 7–16.

[35] Laurens van der Maaten and Geoffrey Hinton. 2008. Viualizing data using t-SNE. *Journal of Machine Learning Research* 9 (2008), 2579–2605.

[36] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NIPS*. 5998–6008.

[37] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. 2017. Graph attention networks. *arXiv preprint arXiv:1710.10903* (2017).

[38] Ke Wang and Xiaojun Wan. 2021. TransSum: Translating aspect and sentiment embeddings for self-supervised opinion summarization. In *Proceedings of ACL-IJCNLP*. 729–742.

[39] Lu Wang and Wang Ling. 2016. Neural network-based abstract generation for opinions and arguments. *arXiv preprint arXiv:1606.02785* (2016).

[40] Xuanhui Wang, Cheng Li, Nadav Golbandi, Michael Bendersky, and Marc Najork. 2018. The LambdaLoss framework for ranking metric optimization. In *Proceedings of CIKM*. 1313–1322.

[41] Yue Wang, Jing Li, Hou Pong Chan, Irwin King, Michael R Lyu, and Shuming Shi. 2019. Topic-aware neural keyphrase generation for social media language. *arXiv preprint arXiv:1906.03889* (2019).

[42] Chuhan Wu, Fangzhao Wu, Tao Qi, and Yongfeng Huang. 2021. Feedrec: News feed recommendation with various user feedbacks. *arXiv preprint arXiv:2102.04903* (2021).

[43] Wenting Xiong and Diane Litman. 2014. Empirical analysis of exploiting review helpfulness for extractive summarization of online reviews. In *Proceedings of COLING*. 1985–1995.

[44] Yang Yang, Junmei Hao, Canjia Li, Zili Wang, Jingang Wang, Fuzheng Zhang, Rao Fu, Peixu Hou, Gong Zhang, and Zhongyuan Wang. 2020. Query-aware tip generation for vertical search. In *Proceedings of CIKM*. 2893–2900.

[45] Qi Zhang, Yang Wang, Yeyun Gong, and Xuan-Jing Huang. 2016. Keyphrase extraction using deep recurrent neural networks on twitter. In *Proceedings of EMNLP*. 836–845.

[46] Xueying Zhang, Yunjiang Jiang, Yue Shang, Zhaomeng Cheng, Chi Zhang, Xiaochuan Fan, Yun Xiao, and Bo Long. 2021. DSGPT: Domain-specific generative pre-training of transformers for text generation in e-commerce title and review summarization. In *Proceedings of SIGIR*. 2146–2150.

[47] Yingyi Zhang, Jing Li, Yan Song, and Chengzhi Zhang. 2018. Encoding conversation context for neural keyphrase extraction from microblog posts. In *Proceedings of NAACL*. 1676–1686.

[48] Yingyi Zhang and Chengzhi Zhang. 2019. Using human attention to extract keyphrase from microblog post. In *Proceedings of ACL*. 5867–5872.