ECAI 2020
G.D. Giacomo et al. (Eds.)

doi:10.3233/FAIA200323

Behavior Based Dynamic Summarization on Product Aspects via Reinforcement Neighbour Selection

Zheng Gao[1] and **Lujun Zhao**[2] and **Heng Huang**[3] and **Hongsong Li**[4] and **Changlong Sun**[5] and **Luo Si**[6] and **Xiaozhong Liu**[7]

Abstract. Dynamic summarization on product aspects, as a newly proposed topic, is an important task in E-commerce for tracking and understanding the nature of products. This can benefit both customers and sellers in different downstream tasks, such as explainable recommendations. However, most existing research works focus on analyzing product static reviews but miss dynamic sentiment changes. In this paper, we propose an innovative multi-task model to sample neighbour products whose information is simultaneously utilized to generate product summarization. In detail, a reinforcement learning approach selects neighbour products from a group of seed products by considering their pairwise similarities calculated from user behaviors. Meanwhile, a generative model helps to summarize product aspects via product descriptive phrases and selected neighbour products' sentimental phrases. To the best of our knowledge, this is the first work that studies dynamic product summarization leveraging user behaviors instead of self-reviews. It means that the proposed approach can naturally address the cold-start scenario where few recent product reviews are available. Extensive experiments are conducted with real-world reviews plus behavior data to validate the proposed method against several strong alternatives.

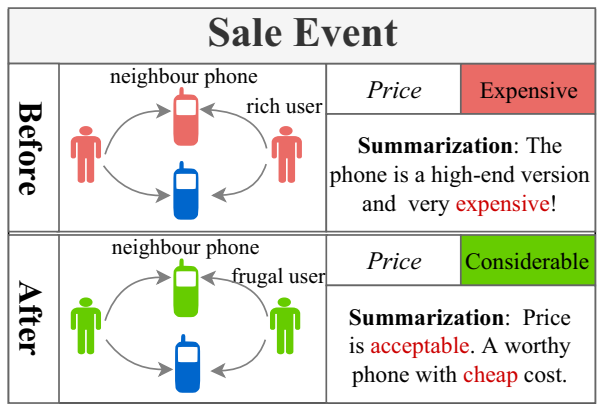


Figure 1: An example to illustrate how to dynamically select neighbour products (red phone → green phone) for depicting current product (blue phone) sentiment change before & after a sale event from user behaviors.

1 Introduction

Understanding product aspect-sentiments and tracking its changes in a timely manner can support better decision-making for commercial purposes, such as enlightening online retailers to make timely sales plans[43]. Therefore, the *Dynamic Summarization on Product Aspects* task is of great importance. It not only indicates products' dynamic aspect-sentiment changes, but also depicts the changes into readable contexts for easier interpretation.

Prior investigations on another similar topic, review summarization, mainly follow Natural Language Generation (NLG) approaches. [31] introduces a new Recurrent Neural Network (RNN) variant that uses gated connections to construct a character-level text generation model; [21] designs a multi-task model to predict rating and generate review summarization simultaneously using pairwise user-product relationship; [36] uses a memory network for review summarization generation. However, all these models are originally designed for static review summarization and take product reviews as model input. They are vulnerable to depict product sentiment changes because of (real-time review) data sparsity. For instance, after investigating 2.16 billion products sold by *Taobao*, a world-leading online shopping website owned by Alibaba, only 0.05% of products are able to gather more than 100 reviews within a three-day window. Thus, review-based approaches are not feasible for dynamic summarizations in large scope because of the lack of instant reviews.

On the other hand, user behavior offers an alternative to address sentiment dynamics. Based on the statistics of the *Taobao* collection, more than 2.53% of products can receive more than 100 multi-type user behaviors (e.g., *'Click'* or *'Purchase'*) within a three-day window where the coverage is 50 times greater than the review scope. Rational Choice Theory [5], on the theory side, proves that user shopping behavior rationality has a coherent relationship with the product peculiarity. As Figure 1 depicted, when a sale event on a high-end phone brings frugal users' instant *clicks*, behavior-based algorithms can immediately consume this information and locate updated neighbour products (red phone → green phone), whose sufficient reviews help to update the product (blue phone) summarization. For review-based approaches, accumulating enough reviews to characterize this dynamic change may take a longer time.

In this paper, instead of review summarization, we aim to generate product aspect summarization in a dynamic manner. They are similar topics but still with huge differences in terms of concept definition and generated summary context. Conceptually, review sum-

[1] Indiana University Bloomington, United States, email: gao27@indiana.edu
[2] Alibaba Group, China, email: ljzhao16@fudan.edu.cn
[3] Red Co.Ltd, China, email:streakly@gmail.com
[4] Alibaba Group, China, email: hongsong.lhs@alibaba-inc.com
[5] Alibaba Group, China, email: changlong.scl@taobao.com
[6] Alibaba Group, China, email: luo.si@alibaba-inc.com
[7] Indiana University Bloomington, United States, email: liu237@indiana.edu

marization reflects customer subjective and personalized expressions on products. While aspect summarization contains objective descriptions only on restricted product aspects. Contextually, review summarization contains more emotional and general terms in a free format, such as 'I love its color so much'. While aspect summarization generates more formal and descriptive expressions, which only focus on specific aspects such as 'price is more expensive than expected'.

Motivated by all mentioned above, we propose a **B**ehavior based **D**ynamic **S**ummarization (BDS) model to accommodate user behavior for dynamic product aspect summarization. [6] proves the user shopping preference is stable in a relatively long-term period, which offers us the theoretical feasibility to learn product behavior representation from user dynamic behavior and consistent shopping preference. The learned representation supports neighbor product selection from a group of seed products with abundant instant reviews (**Task 1**) and meanwhile implicitly helps to generate aspect summarization from product own descriptive phrases and neighbour products' filtered sentimental phrases (**Task 2**). As both user behavior and seed products' instant reviews are changed across time, the selected neighbour products and generated summarizations are associated with changes as well. The contribution of this work is threefold:

- To the best of our knowledge, this is the first effort to leverage user behavior for dynamic summarization on product aspects. This work pioneers behavior-based summarization investigation.
- In our model, a reinforcement learning approach learns the sampling strategy on seed products with rewards from both sentiment (calculated from behavior-to-sentiment prediction) and semantic (calculated from summarization generation) viewpoints. When generating product aspect summarization, our model does not require target product's reviews as model input, which is able to solve review sparseness, even zero-review, problem.
- Experiments on a large E-commerce dataset show that our proposed model significantly outperforms the baselines from both automatic and human perspectives. Extensive studies also prove the efficacy of each model input component.

2 BDS Model

The overall model architecture is sketched in Figure 2. Our final goal is to generate product dynamic aspect summarizations using behavior data instead of product reviews. To optimize BDS model, the involved data are categorized into two groups: training & validation data and seed product data. The training & validation data contains products with both sufficient user behavior and temporal reviews in an individual time period. While seed product data are products required to have sufficient reviews across all time periods. The training & validation data helps to calculate dynamic product behavior presentations (*Section 2.2*) and pretrain product sentiment prediction model (*Section 2.3.1*). Subsequently, a multi-task model dynamically selects neighbour products (*Section 2.3.2*) from seed products whose sentimental phrases filtered from reviews contribute to generate product aspect summarization (*Section 2.4*).

2.1 Data Prerequisite

Before optimizing BDS model, four following types of information need to be clarified and extracted as data prerequisite:

Product Sentiment Distribution: AliNLP[8], a paid NLP service by Alibaba, is particularly developed to analyze the E-commerce reviews' sentiment polarity on aspects such as '*Quality*' or '*Fitness*'. We use its sentiment analyzer to label and rate the sentiment of a review towards a specific aspect as 'None (0.5)', 'Negative (0)' or 'Positive (1)'. Each aspect sentiment score of a given product is calculated as the weighted average score of all three labels appeared in its eligible reviews (with more than ten words). Product sentiment distribution is a vector of all aspect sentiment scores normalized by dividing its sum, which is regarded as the ground truth for Sentiment Prediction Pretraining (Section 2.3.1). In this paper, four aspects are selected to construct product sentiment distribution, which will be illustrated in the Experiments (Section 3).

Product Descriptive Phrase: Each product is associated with a description profile. After applying AliNLP Named Entity Recognizer (NER) on their profiles, we only keep the noun phrases to characterize related products.

Review Sentimental Phrase: The AliNLP sentiment analyzer can also extract sentimental phrases related to specific aspects from seed product reviews, i.e., 'poor quality' is a sentimental phrases of '*Quality*' aspect. Later on, these filtered sentimental phrases will be concatenated with product descriptive phrases as the model input for aspect summarization (Section 2.4).

Seed Products: The top products having the most sufficient reviews across all t time periods are regarded as seed products. Their aspect sentiment distributions are pre-calculated directly from reviews in each time period, which are later used to guide neighbour product selection (Section 2.3).

2.2 Dynamic Product Behavior Representation

Matrix factorization is utilized on dynamic user behaviors to learn product behavior representation across all time periods. In the initial time period, we learn both user and product m types of behavior representations via related user behavior $\{B_1^{(0)}, ..., B_m^{(0)}\}$, where a behavior type can refer to '*Click*' or '*Purchase*', etc. The i_{th} type of user behavior $B_i^{(0)}$ is in a format of sparse matrix where each row denotes a user and each column denotes a product. The data points in $B_i^{(0)}$ denote users' i_{th} type of behaviors on products. Considering all potential methods listed in *Section 3.2*, we empirically select Probabilistic Matrix Factorization (PMF) to decompose $B_i^{(0)}$ to user representation matrix $U_i^{(0)}$ and product representation matrix $P_i^{(0)}$ by satisfying the following equation:

$$B_i^{(0)} = U_i^{(0)}(P_i^{(0)})^\intercal, i = 1, ..., m \tag{1}$$

As user shopping preference is consistently stable [6], once user representation $U_i^{(0)}$ is calculated from the initial period, it remains unchanged and bridges product dynamic behavior representation in subsequent time periods.

In the t_{th} time period, the matrix form of Least Squares Approximation [26] helps to calculate the i_{th} type of product behavior representation $P_i^{(t)}$ given related user behavior $B_i^{(t)}$ and user consistent representation $U_i^{(0)}$:

$$\begin{aligned} & B_i^{(t)} = U_i^{(0)}(P_i^{(t)})^\intercal, i = 1, ..., m \\ \Rightarrow\ & (U_i^{(0)})^\intercal(B_i^{(t)} - U_i^{(0)}(P_i^{(t)})^\intercal) = 0 \\ \Rightarrow\ & (U_i^{(0)})^\intercal U_i^{(0)}(P_i^{(t)})^\intercal = (U_i^{(0)})^\intercal B_i^{(t)} \\ \Rightarrow\ & (P_i^{(t)})^\intercal = ((U_i^{(0)})^\intercal U_i^{(0)})^{-1}(U_i^{(0)})^\intercal B_i^{(t)} \\ \Rightarrow\ & P_i^{(t)} = (B_i^{(t)})^\intercal U_i^{(0)}((U_i^{(0)})^\intercal U_i^{(0)})^{-1} \end{aligned} \tag{2}$$

[8] English Tutorial: https://www.alibabacloud.com/help/product/57736.html
Chinese Tutorial: https://data.aliyun.com/product/nlp

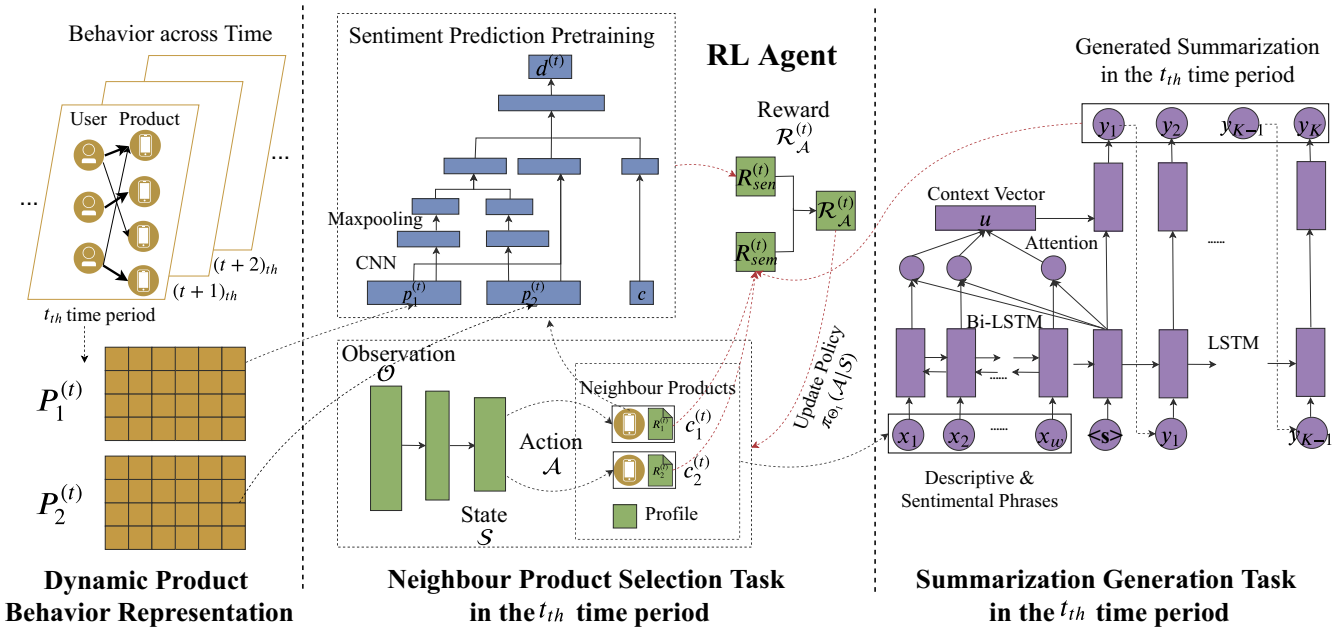


Figure 2: The overall architecture of BDS model in the t_{th} time period. Each color refers to an individual model component. Dynamic Product Behavior Representation and Sentiment Prediction Pretraining are optimized before training the main multi-task model (Neighbour Product Selection & Summarization Generation). Red dashed lines show the workflows to calculate RL reward and in return to optimize RL policy. Data Prerequisite steps are omitted here for simplicity but are illustrated in detail in the main paper.

2.3 Neighbour Product Selection Task

2.3.1 *Sentiment Prediction Pretraining*

In the t_{th} time period, we combine product behavior representation and category to estimate product aspect sentiment distribution via a hybrid CNN-MLP (Convolutional Neural Network and Multilayer Perceptron) approach. Let c be the category of a target product, stored originally as one-hot embedding. To better represent its enriched information, we first apply an one-layer MLP with $\tanh(\cdot)$ activation function to convert it as a dense vector v^c.

$$v^c = \tanh(W^c c + b^c) \tag{3}$$

where W^c and b^c denote the weight matrix and bias respectively.

For the same product, we can also obtain its multi-type behavior representation $\{p_1^{(t)}, ..., p_m^{(t)}\}$, where $p_m^{(t)} \in P_m^{(t)}$ denotes its m_{th} type behavior representation. As CNN kernel can help to filter out the most important dimensions (local features) from a vector, a CNN layer $\mathrm{cnn}(\cdot)$ with Max pooling mechanism $\mathrm{max\_pooling}(\cdot)$ is applied to capture product i_{th} type local feature representation $l_i^{(t)}$.

$$l_i^{(t)} = \mathrm{max\_pooling}(\tanh(\mathrm{cnn}(p_i^{(t)}))), i = 1, ..., m \tag{4}$$

Subsequently, we average all m types of local feature representation to a single vector $l^{(t)}$. Similarly, we calculate the average product behavior vector $p^{(t)}$ as global feature representation of m types of behaviors:

$$\begin{aligned} l^{(t)} &= \frac{1}{m}\sum_{i=1}^{m} l_i^{(t)} \\ p^{(t)} &= \frac{1}{m}\sum_{i=1}^{m} p_i^{(t)} \end{aligned} \tag{5}$$

In the end, a product has three types of information representation, including global feature representation $p^{(t)}$, local feature representation $l^{(t)}$, and category representation v^c. We concatenate all these information to calculate the estimated product sentiment distribution $d^{(t)}$ over all pre-selected aspects via one layer MLP normalized by $\mathrm{softmax}(\cdot)$ function:

$$d^{(t)} = \mathrm{softmax}(W^d[p^{(t)}, l^{(t)}, v^c] + b^d) \tag{6}$$

where W^d and b^d are related weight matrix and bias. $[,]$ denotes the concatenation operation.

This pretraining process is optimized by minimizing the cross entropy between the estimated product sentiment distribution $d^{(t)}$ and the actual product sentiment distribution calculated from product reviews (Section 2.1). This optimization step has to be done ahead of the main multi-task model introduced in the following sections.

2.3.2 *Reinforcement Neighbour Selection*

In the t_{th} time period, among h seed products with sufficient reviews, a policy gradient approach [30] learns an action to select s neighbour products $\mathcal{A} = \{\alpha_1^{(t)}, \alpha_2^{(t)}, ..., \alpha_s^{(t)}\}$ out of h candidates where $\alpha_s^{(t)}$ denotes the s_{th} selected neighbour product. As we do not consider the sampling sequence on neighbour products, the reinforcement approach is a one-step Markov Decision Process (MDP) with single state $\mathcal{S}$ and single action $\mathcal{A}$.

Assume there are n products in total across all time periods, given a target product, the initial observation $\mathcal{O}$ is the product itself, represented as a one-hot embedding $\in \mathbb{R}^n$. The state $\mathcal{S}$ is learned via a two-layer Multilayer Perception (MLP) on the initial observation $\mathcal{O}$:

$$\mathcal{S} = \mathrm{softmax}(W_2 \tanh(W_1 \mathcal{O} + b_1)) \tag{7}$$

where W_1,W_2 and b_1 denote related weight matrices and bias, respectively.

The learned state $\mathcal{S} \in \mathbb{R}^h$ is the selection probability distribution over h seed products. Assuming an action is taken to sample s neighbour products with related probability weights $\{\omega_1^{(t)}, \omega_2^{(t)}, ..., \omega_s^{(t)}\} \in \mathcal{S}$, the sampling policy $\pi_{\Theta_1}(\mathcal{A}|\mathcal{S})$ can be therefore calculated as:

$$\pi_{\Theta_1}(\mathcal{A}|\mathcal{S}) = s! \prod_{i=1}^{s} \omega_i^{(t)} \tag{8}$$

Θ_1 denote the parameters to be learned. The factorial of s ($s!$) denotes the number of permutations for the selected neighbour products as the neighbour products are sequence insensitive. $\prod_{i=1}^{s} \omega_i^{(t)}$ is the generative probability of each permutation.

To assess the fitness of the s selected neighbour products for the target product, we design two dynamic rewards: a sentiment reward to measure the aspect-sentiments similarity, and a semantic reward to calculate the content similarity between s neighbour products and the target product.

Sentiment Reward: In the t_{th} time period, we estimate the target product's sentiment distribution as $d_a^{(t)}$ by the Sentiment Prediction Pretraining (Section 2.3.1). The sentiment distributions of all its selected neighbour products $\{d_1^{(t)}, ..., d_s^{(t)}\}$ are calculated directly from their reviews (Section 2.1). To evaluate pairwise distribution similarity, Pearson correlation [4] calculates the sentiment reward $\mathcal{R}_{sen,i}^{(t)}$ of the i_{th} selected neighbour product $\alpha_i^{(t)}$ as follows:

$$\mathcal{R}_{sen,i}^{(t)} = \frac{\mathbb{E}[(d_a^{(t)} - \mu(d_a^{(t)}))(d_i^{(t)} - \mu(d_i^{(t)}))]}{\sigma(d_a^{(t)})\sigma(d_i^{(t)})}, i = 1, ..., s \tag{9}$$

where $\mathbb{E}(\cdot)$ denotes the expectation, $\mu(\cdot)$ denotes the mean and $\sigma(\cdot)$ denotes the standard deviation. Larger similarity score offers a higher reward to the related neighbour product.

Semantic Reward: In the t_{th} time period, the semantic reward of neighbour products is measured by the accuracy of generated product summarization. In this paper, we use the word level Jaccard Similarity as the indicator to calculate the semantic reward $\mathcal{R}_{sem}^{(t)}$ for all selected neighbour products, which contains two parts: the averaged Jaccard Similarity between all neighbour product original reviews and real product summarization, and the Jaccard Similarity between generated product summarization and actual product summarization:

$$\mathcal{R}_{sem}^{(t)} = \frac{1}{s} \sum_{i=1}^{s} \frac{|R_i^{(t)} \cap Y^{(t)}|}{|R_i^{(t)} \cup Y^{(t)}|} + \frac{|\hat{Y}^{(t)} \cap Y^{(t)}|}{|\hat{Y}^{(t)} \cup Y^{(t)}|} \tag{10}$$

where $R_i^{(t)}$ denotes all original reviews of the i_{th} neighbour product $\alpha_i^{(t)}$, $\hat{Y}^{(t)}$ denotes the generated summarization of target product, and $Y^{(t)}$ denotes its actual summarization. The total reward $\mathcal{R}_{\mathcal{A}}^{(t)}$ of neighbour products is the weighted sum between sentiment reward and semantic reward controlled by a weighting factor γ:

$$\mathcal{R}_{\mathcal{A}}^{(t)} = \sum_{i=1}^{s} \omega_i^{(t)} \mathcal{R}_{sen,i}^{(t)} + \gamma \mathcal{R}_{sem}^{(t)} \tag{11}$$

Task Optimization: We use policy gradient method to optimize the sampling policy, aiming to maximize the expected total reward for neighbour products. The expected reward $\mathcal{J}_{sel}(\Theta_1)$ in the t_{th} time period is:

$$\mathcal{J}_{sel}(\Theta_1) = \mathbb{E}_{\mathcal{A} \sim \pi_{\Theta_1}(\mathcal{A}|\mathcal{S})}[\mathcal{R}_{\mathcal{A}}^{(t)}] \tag{12}$$

Then, the gradient is estimated using the likelihood ratio trick [30]:

$$\begin{aligned} \nabla_{\Theta_1} \mathcal{J}_{sel}(\Theta_1) &= \nabla_{\Theta_1} \sum_{\mathcal{A}} \pi_{\Theta_1}(\mathcal{A}|\mathcal{S}) \mathcal{R}_{\mathcal{A}}^{(t)} \\ &\approx \frac{1}{N} \sum_{i=1}^{N} \nabla_{\Theta_1} log \pi_{\Theta_1}(\mathcal{A}_i|\mathcal{S}) \mathcal{R}_{\mathcal{A}_i}^{(t)} \end{aligned} \tag{13}$$

where $\mathcal{A}_i$ denotes the i_{th} of N randomly sampled actions (selecting s neighbour products from h seed products).

2.4 Summarization Generation Task

In the t_{th} time period, the filtered neighbour product sentimental phrases together with product own descriptive phrases are concatenated into a sequence $X^{(t)} = \{x_1, ..., x_w\}$. It is used as the input of Neural Machine Translation (NMT) model [2] to generate aspect summarization sequence $Y^{(t)} = \{y_1, ..., y_k\}$. w and k denote the input and output sequence length, respectively. Filtering out most of emotional and other irrelevant words in advance can better map the input to aspect oriented summarizations instead of subjective review summaries.

The input sequence $X^{(t)} = \{x_1, ..., x_w\}$ is fed one-by-one into the encoder (a single-layer bidirectional LSTM), producing a sequence of encoder hidden states $\{e_1, ..., e_w\}$. In decoding step i, the decoder (a single-layer unidirectional LSTM) has a decoder hidden state h_i. Its context vector u_i is generated via an Attention mechanism [33] on all encoder hidden states and current decoder hidden state:

$$\begin{aligned} a_{ij} &= \text{attention}(h_i, e_j), j = 1, ..., w \\ a_{ij}^* &= \frac{\exp(a_{ij})}{\sum_{k=1}^{w} \exp(a_{ik})} \\ u_i &= \sum_{j=1}^{w} a_{ij}^* e_j \end{aligned} \tag{14}$$

where a_{ij} denotes the attention weight of encoder hidden state e_j. a_{ij}^* is the normalized weight by Softmax function. The weighted sum of all encoder hidden states, u_i, is the context vector for current step i, reflecting the auxiliary information from input sequences.

A one-layer MLP is subsequently utilized on the combination of context vector u_i and decoder hidden state h_i to generate the vocabulary probability distribution P_{vocab}:

$$P_{vocab} = \text{softmax}(W^o[u_i, h_i] + b^o) \tag{15}$$

where W^o and b^o are related weight and bias.

In decoding step i, the generation loss for target word y_i is its negative log likelihood, $-logP_{vocab}(y_i)$. The overall generation loss $\mathcal{J}_{gen}(\Theta_2)$ is the average of all k step generation losses and Θ_2 are all related parameters to be optimized.

$$\mathcal{J}_{gen}(\Theta_2) = \frac{1}{k} \sum_{i=1}^{k} -logP_{vocab}(y_i) \tag{16}$$

In the multi-task model, $\mathcal{J}_{sel}(\Theta_1)$ and $\mathcal{J}_{gen}(\Theta_2)$ both need to be minimized during the model training process. Each task is learned separately and alternately after taking a certain number of training data batches in their optimization processes.

3 Experiments

3.1 Dataset

From *Taobao*, a world-leading E-commerce website owned by Alibaba, we collect user behavior and product reviews during the period from Apr/12/2018 to Jul/10/2018.the raw dataset is split into two parts: the first two-week data, as the initial time period t_0, helps to generate user consistent shopping preference (*Section 2.2*). For the rest data, we empirically set fifteen days as the time window to split it evenly into five consecutive time periods ($t_1 \sim t_5$). In each time period, four types of information need to be prepared and calculated in advance to support model training, including multi-type user behavior, product profile, product sentiment distribution, and product ground truth summarization:

First, multi-type user behavior is used for generating dynamic product behavior representation (*Section 2.2*). Three types of user behavior are considered in this paper including '*Click*', '*Add Cart*' and '*Purchase*'.

Second, from product profile, product category is encoded as one-hot embedding for sentiment distribution prediction (*Section 2.3*). Descriptive phrases are extracted from product description profiles to support aspect summarization generation (*Section 2.4*).

Third, product sentiment distribution is calculated from reviews (*Section 2.1*) as the ground truth of sentiment prediction pretraining (*Section 2.3*). We consider four common aspects including '*Quality*', '*Cost-performance Ratio*', '*Fitness*' and '*Material*'.

Fourth, in *Taobao*, a user can rate reviews with thumbs-up or thumbs-down signal. For each product in the training data, we firstly select its top ten reviews relevant to the four picked aspects and with the largest number of thumbs-ups. After that, using AliNLP sentiment analyzer, we can locate and filter out the aspect-related sentences from the reviews as the ground truth for aspect summarization generation (*Section 2.4*).

In total, the whole dataset contains 125,598 products, 51,366 users and 108,749,788 multi-type behaviors. We use 80% of the data for training, 10% for validation and 10% for testing.

3.2 Baselines and Settings

As the main contribution of this paper lies in the reinforcement neighbour product selection task, five baselines are chosen to compare from neighbour product selection viewpoint. **1) Title Similarity** (TS): Neighbour products are selected with the shortest Levenshtein Distance [26] on titles. **2) Random**: Neighbour products are randomly selected. **3) PMF**: PMF [25] firstly learns product embeddings via matrix factorization. Neighbour products are selected with the highest cosine similarity score on product embeddings. **4) GBPR**: GBPR [28] uses a Bayesian based collaborative filtering method to assign user preferences on products. Neighbour products are selected with the most similar user preferences. **5) EALS**: EALS [20] is a fast matrix factorization approach which learns product embeddings. Neighbour products are selected with the highest cosine similarity score on product embeddings. After the neighbour products are selected, their reviews' sentimental phrases together with product own descriptive phrases are utilized to generate aspect summarization via the same NMT model struture [2] as our BDS model. We intentionally apply the same generative model so as to compare the effectiveness of neighbour products selection in different models.

To assess the usefulness of neighbour product information, our model is compared with another two generative models only leveraging product own metadata. **6) Raw Title** (RT) uses the product titles and **7) Title-Review** (TR) uses the concatenation of product title and sparse reviews as the input of the same NMT model to generate aspect summarization.

Mini-batch (*size = 20*) Adam SGD optimizer is used to train our model for 100 epochs. Learning rate is 0.01. Dimension of product & user representation is 300 (*Section 2.2*). $s = 5$ neighbour products are selected from $h = 100$ seed products (*Section 2.3*). Vocabulary size in summarization generation (*Section 2.4*) is *15K*. Other parameter details will be offered once the paper gets published.

3.3 Evaluation Metrics

In this paper, we report the model performance via the following metrics from both automatic and human perspectives. Automatic metrics evaluate the accuracy of the generated summarizations by objectively calculating the correctness of predicted words. While human metrics evaluate the semantic quality of generated summarizations by subjectively considering their sentence readability.

- **ROUGE** (RG): [22] evaluates text summarization quality by comparing the overlap between generated sequences and the ground truth. We report RG-1, RG-2 and RG-L in this paper.
- **METEOR**: [10] is the harmonic mean of generated summarizations' unigram precision and recall. It offers stemming and synonymy matching along with standard exact word matching.
- **Human Evaluation**: We generate summaries of 200 random products by each of the seven baselines and our model. To compare BDS model with each baseline, three human votes are collected from a crowd-sourcing platform for each pair of product summaries ($200 * 7 * 3 = 4,200$ human judgements). People are required to vote the one with more comprehensive information and better sentence structure. We define *winning time rate* (WTR) and *winning count rate* (WCR) as two human evaluation metrics. Given a pair of model A and model B, the WTR of A is the ratio of winning products for A and the WCR of A is the ratio of winning votes for A. For instance, given two products α and β, if method A gets two votes for α and one vote for β, its WTR is $(1+0)/(1+1) = 0.5$ and WCR is $(2+1)/(3+3) = 0.5$.

3.4 Results

As neighbour product selection (Task 1) is an unsupervised approach whose ultimate goal is to support product summarization (Task 2), we only report the experimental results on Task 2 to demonstrate our model's superiority over the baselines across different timestamps. The efficacy of each input component is also presented here.

3.4.1 Automatic Evaluation

We run our model ten times and report the average evaluation results in the left part of Table 1. To verify our model's superiority, we calculate the performance differences between our model and each baseline on each automatic metric for all the ten runs, and apply a t-test [11] on the ten differences to check whether the performance difference is significant.

RT performs the worst as the product title contains limited and static information to reveal product sentiment dynamics. From TR results, adding sparse reviews can improve model performance, but is still worse than the rest approaches, which strongly indicates the effectiveness of using neighbour product reviews for generating aspect summarization. Most of neighbour selection based baselines

have roughly similar results except TS model, meaning that behavior based is better than content based neighbour selection. Surprisingly, Random model can achieve a relatively satisfying result. One possible reason is that because most of reviews in online shopping websites are positive, randomly sampled neighbour product might receive reviews containing relevant sentimental phrases. Howsoever, our BDS model outperforms all baselines significantly ($p<0.01$) on all metrics, demonstrating the superiority of our proposed reinforcement neighbour selection.

Model	Automatic				Human	
	RG-1	RG-2	RG-L	METEOR	WTR	WCR
RT	36.19	10.01	27.95	16.05	0.00	0.01
TR	43.05	19.08	33.10	19.24	0.07	0.13
TS	42.97	18.85	32.85	19.34	0.13	0.24
Random	44.21	19.58	33.98	19.86	0.03	0.13
PMF	45.72	20.25	35.21	20.80	0.14	0.28
GBPR	45.09	19.67	34.55	20.36	0.32	0.39
EALS	44.66	19.50	34.28	20.32	0.08	0.17
BDS	**51.11***	**23.55***	**39.86***	**22.99***	**0.89**	**0.81**

Table 1: Automatic & human evaluation results of our model compared with baselines. Symbol '*' highlights the cases where our model significantly beats all baselines with p value smaller than 0.01.

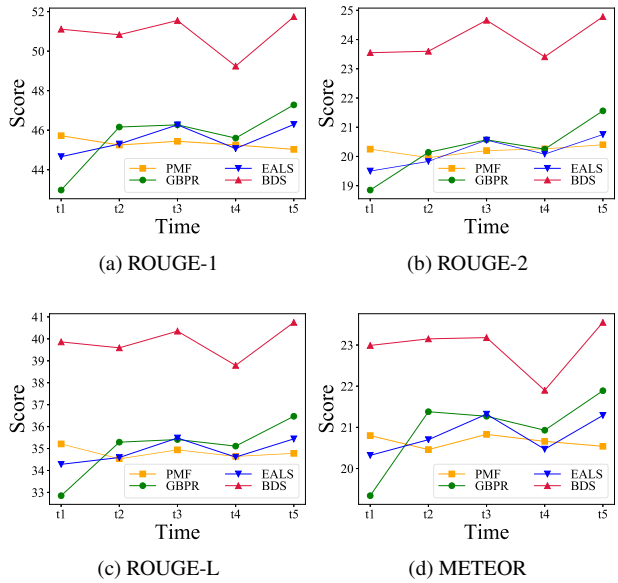
3.4.2 *Human Evaluation*

The right part of Table 1 reports the human evaluation result for all the models. Higher WTR and WCR scores indicate the related model can generate better structured summaries from human perspective. For each baseline, WTR and WCR scores are the pairwise comparison results with the BDS model. RT model performs the worst. And content based methods (RT and TR) also perform worse than neighbour selection based methods in general. GBPR performs much better than the rest baselines. It beats our model in roughly 30% of summary pairs. The reported two scores of our model are the average of pairwise comparison results with all baselines, which shows that our model can beat other baselines on 90% of all summary pairs.

Both automatic and human evaluation results demonstrate content based baselines perform worse than neighbour selection based baselines. However, there are still some inconsistencies between their evaluation results. Although GBPR has similar performance with the rest of neighbour based baselines on automatic metrics, it unexpectedly outperforms on human metrics, which indicates how to organize sequences has huge impact on summary semantic quality.

3.4.3 *Dynamic Performance*

To better present our model dynamic performance, we visualize the automatic evaluation results of our model as well as the best three baselines (PMF, EALS and GBPR) in all five consecutive time periods, shown in Figure 3. The three best baselines are all neighbour selection approaches. Across all time periods, all four model performances are relatively consistent and follow similar trend. Their performances go down a little bit in the fourth time period but raise up immediately in the next time period. Among three baselines, GBPR achieves the best performance result over the rest two. It does not perform well in the beginning, but keeps growing and beats the rest baselines in later time periods. In general, the performances of three baselines are not far away from each other, especially from time period t_2 to t_4. Shown in Figure 3, their plotted lines are basically mingled together. However, our model achieves a far better evaluation



(a) ROUGE-1 (b) ROUGE-2

(c) ROUGE-L (d) METEOR

Figure 3: Automatic evaluation results of our model compared with the Top 3 baselines in five consecutive time periods ($t_1 \sim t_5$).

result than baselines. Its plotted lines (red lines) are always significantly above the rest three lines in terms of all four evaluation metrics. Moreover, model performances on the four reported metrics are with similar trend. And the three ROUGE based metrics are with an even more similar trend than METEOR.

3.4.4 *Input Component Evaluation*

As aforementioned, our model requires three types of product input information: user behavior, product category, and profile descriptive phrases. To examine whether all involved information are effective, we conduct an extensive study by removing each type of information iteratively while holding the rest information fixed. Table 2 shows the performance difference between the models with truncated input and original BDS model. In detail, removing user behavior (product category) means that only product category (user behavior) is used for sentiment prediction pretraining. Removing product profiles means that only neighbour product sentimental phrases contribute to summarization generation.

Model	RG-1	RG-2	RG-L	METEOR
– Behavior	-14.21	-12.63	-11.37	-5.72
– Category	-4.74	-2.94	-4.27	-1.65
– Profile	-5.78	-3.77	-5.33	-2.34

Table 2: Performance differences between the model with truncated input and original BDS model. '–' means removing related input component from our model.

From Table 2, removing related input component always leads a decrease on model performance, which indicates that all the three types of input component are useful in BDS model. Compared with product category and descriptive phrases, user behavior obviously has the most influential impact because removing it causes the largest drop in model performance over all four reported metrics. Surprisingly, it declines the model performance down close to the worst

baseline, RT. It explicitly verifies the importance of user behavior for product summarization generation. Moreover, the performance decrease by removing profile descriptive phrases is larger than removing product category embedding. The reason might be that profile descriptive phrases are directly used for summarization generation. While product category contributes to the sentiment prediction pretraining, which only has implicit impact and is not able to directly reflect dynamic aspect sentiment changes.

3.4.5 *Case Study*

We conduct a case study in Table 3 on an actual dress to show how our model summarize its aspect-sentiment changes in a timely manner. In this case, we only care about sentiment changes on product material and demonstrate material-related content from the full generated summaries. In the real world, the sentiment of the dress material goes from positive to negative (concluded from summarization) due to its manufacturer's counterfeit (reported by customers in time t_2). Our model is able to detect this aspect-sentiment change from its updated customer shopping behaviors and dynamically locate neighbour products with similar issues (like material problems). From the result shown in Table 3, the generated summaries on 'Material' aspect supports our model effectiveness. Moreover, the generated summaries solely contain objective descriptions instead of emotional expressions used in personal reviews, which shows a more formal and aspect-concentrated way of description than summarized reviews.

Time	Ground Truth	BDS
t_1	**Material : Positive** **Summarization:** The material of the dress is super nice and soft to wear. It is made of cotton and touches like a high-end dress. A perfect gift for aged women.	**Material : Positive** **Summarization:** The dress has good material . It is made of cotton and touches very soft. A perfect gift for mums who have high-standard requirement on dress material.
t_2	**Material : Negative** **Summarization:** The dress material is **not as good as promised**. It is **not good** for aged women with high requirements. It touches as **cheap material** and smells weird.	**Material : Negative** **Summarization:** The **material is terrible**. It touches like **carded yarn with much cheaper material** as promised for mums. It has a weird smell when wearing it.

Table 3: A real case to show our model's generated dynamic sentiment summarization for a dress on product material. The green color indicates positive words. While red color indicates negative words.

4 Related Works

Behavior Analysis: Revealing the coupling between user behavior and product peculiarity has been explored for years. By applying matrix factorization techniques, user-product behavior matrix can be decomposed into a user matrix and a product matrix where each product is represented as a dense vector [25]. BPR [29] proposes a Bayesian approach to learn personalized user preferences on products. In this approach, a product's neighbours have similar user preferences with itself. GBPR [28] extends from BPR and adds group preference to predict the relationship between users and products. EALS [20] designs a fast matrix factorization method by weighting the missing data based on product popularity. TrustSVD [19] selects trust data from both explicit and implicit user feedbacks to calculate user and product representations. [44] develops a complicated localized matrix factorization method to learn product representations based on matrix block diagonal forms. Deep Matrix Factorization [40] is proposed to consider both user and product behavior as the input to predict the user-product pairwise relationship, which derives a series of follow-up works [32, 7].

Summarization Models: Existing works for summarization are either data-to-text or text-to-text approaches [35]. Data-to-text models mostly use structured data as input to generate summaries. [42] uses product aspect-sentiments to summarize product reviews with a hierarchical-structured RNN model. [12] also utilizes product aspect attributes to generate associated reviews with an encoder-decoder framework by conducting the combination of user, product and rating information. [39] offers a practical guide on how to efficiently process such data and train model. [8] learns structured data embedding and uses a Copy mechanism to avoid generating repetitive content.

The input of text-to-text models is usually sequential reviews. [31] firstly uses a character-level RNN model to generate text reviews. After that, [18] applies Long Short-term Memory (LSTM) to improve the performance of generated summaries. [34] uses an attention mechanism so that the model can absorb information from multiple text units. [17] proposes a CNN model to learn a sequence to sequence mapping relationship from reviews. CF-GCN [27] jointly performs recommendation and review generation by combining collaborative filtering and LSTM-based generative models. Similarly, [21, 36] both predict product ratings and generate summarizations via a gated RNN model.

Dynamic Models: A few models consider time impact on products for either review summarization [1] or sentiment analysis [15, 13]. These models play a vital role especially in e-commerce scenarios [14, 16]. ETTS [41] considers time stamped sequences for review summarization. GCN [23] proposes a character level RNN to generate personalized reviews capturing complex product sentiment dynamics. [24] develops a novel semi-supervised method to simultaneously solve sparseness problem on dynamic rating prediction task. And [3] leverages collaborative filtering techniques to track product temporal aspect-sentiments via Singular Value Decomposition (SVD). [38, 37] apply LSTM mechanism in a RNN model to track the temporal dynamics of product aspect-sentiments in an auto-regressive way. [9] designs a time-aware gated recurrent unit (T-GRU) to generate explanations for personalized recommendation.

5 Conclusion

By leveraging multi-type user behaviors instead of sparse reviews, we propose a multi-task model to solve an innovative dynamic summarization task for product aspects. Extensive experiments show our model is consistently promising and significantly outperforms the baselines. Being the first study on this newly proposed task, we aim to explore the relationship between user behavior and product summarization so as to address the cold-start problem, i.e., products without any recent reviews. As a result, the proposed model can cover a much larger scope in the E-commerce ecosystem while enabling explainable sentiment analysis on products. As the generated summarization is sensitive to customers, we never want to make up 'fake reviews' to mislead them. Instead, the summarization should only be provided to online sellers as auxiliary information. In our current model, both product behavior representation and behavior-to-sentiment pretraining need to be learned apart from the multi-task model. In the future, we plan to integrate all separated segments into the main model to achieve joint training. Besides, as the reinforcement neighbour product selection task contributes our major novelty, all compared baselines are intentionally chosen from models aiming at neighbour selection. For the next step, we will put more efforts on exploring the influence of summarization generation task by involving other generative models such as Pointer Network and CopyNet.

REFERENCES

[1] Romi Akpala. Dynamic predefined product reviews, August 13 2019. US Patent 10,380,656.

[2] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio, 'Neural machine translation by jointly learning to align and translate', *arXiv preprint arXiv:1409.0473*, (2014).

[3] Cigdem Bakir, 'Collaborative filtering with temporal dynamics with using singular value decomposition', *Tehnički vjesnik*, **25**(1), 130–135, (2018).

[4] Christopher M Bishop, *Pattern recognition and machine learning*, springer, 2006.

[5] Lawrence E Blume and David Easley, 'Rationality', *The new Palgrave dictionary of economics*, **6**, 884–893, (2008).

[6] Roy Brouwer, Ivana Logar, and Oleg Sheremet, 'Choice consistency and preference stability in test-retests of discrete choice experiment and open-ended willingness to pay elicitation formats', *Environmental and Resource Economics*, **68**(3), 729–751, (2017).

[7] Hung-Hsuan Chen, 'Behavior2vec: Generating distributed representations of users' behaviors on products for recommender systems', *ACM Transactions on Knowledge Discovery from Data (TKDD)*, **12**(4), 43, (2018).

[8] Shuang Chen, 'A general model for neural text generation from structured data', *E2E NLG Challenge System Descriptions*, (2018).

[9] Xu Chen, Yongfeng Zhang, and Zheng Qin, 'Dynamic explainable recommendation based on neural attentive models', in *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 53–60, (2019).

[10] Michael Denkowski and Alon Lavie, 'Meteor universal: Language specific translation evaluation for any target language', in *Proceedings of the ninth workshop on statistical machine translation*, pp. 376–380, (2014).

[11] Ben Derrick, Deirdre Toher, and Paul White, 'How to compare the means of two samples that include paired observations and independent observations: A companion to derrick, russ, toher and white (2017)', *The Quantitative Methods in Psychology*, **13**(2), 120–126, (2017).

[12] Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke 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*, volume 1, pp. 623–632, (2017).

[13] Zheng Gao and Rui Bi, 'University of pittsburgh at trec 2014 microblog track', Technical report, PITTSBURGH UNIV PA SCHOOL OF INFO SCIENCES, (2014).

[14] Zheng Gao, Lin Guo, Chi Ma, Xiao Ma, Kai Sun, Hang Xiang, Xiaoqiang Zhu, Hongsong Li, and Xiaozhong Liu, 'Amad: adversarial multiscale anomaly detection on high-dimensional and time-evolving categorical data', in *Proceedings of the 1st International Workshop on Deep Learning Practice for High-Dimensional Sparse Data*, pp. 1–8, (2019).

[15] Zheng Gao and John Wolohan, 'Fast nlp-based pattern matching in real time tweet recommendation.'.

[16] Zizhe Gao, Zheng Gao, Heng Huang, Zhuoren Jiang, and Yuliang Yan, 'An end-to-end model of predicting diverse ranking on heterogeneous feeds.', (2018).

[17] Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin, 'Convolutional sequence to sequence learning', *arXiv preprint arXiv:1705.03122*, (2017).

[18] Alex Graves, 'Generating sequences with recurrent neural networks', *arXiv preprint arXiv:1308.0850*, (2013).

[19] Guibing Guo, Jie Zhang, and Neil Yorke-Smith, 'Trustsvd: Collaborative filtering with both the explicit and implicit influence of user trust and of item ratings.', in *AAAI*, volume 15, pp. 123–125, (2015).

[20] Xiangnan He, Hanwang Zhang, Min-Yen Kan, and Tat-Seng Chua, 'Fast matrix factorization for online recommendation with implicit feedback', in *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 549–558. ACM, (2016).

[21] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam, 'Neural rating regression with abstractive tips generation for recommendation', in *Proceedings of the 40th International ACM SIGIR conference on Research and Development in Information Retrieval*, pp. 345–354. ACM, (2017).

[22] Chin-Yew Lin, 'Rouge: A package for automatic evaluation of summaries', *Text Summarization Branches Out*, (2004).

[23] Zachary C Lipton, Sharad Vikram, and Julian McAuley, 'Generative concatenative nets jointly learn to write and classify reviews', *arXiv preprint arXiv:1511.03683*, (2015).

[24] Cheng Luo, Xiongcai Cai, and Nipa Chowdhury, 'Self-training temporal dynamic collaborative filtering', in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pp. 461–472. Springer, (2014).

[25] Andriy Mnih and Ruslan R Salakhutdinov, 'Probabilistic matrix factorization', in *Advances in neural information processing systems*, pp. 1257–1264, (2008).

[26] Nasser M Nasrabadi, 'Pattern recognition and machine learning', *Journal of electronic imaging*, **16**(4), 049901, (2007).

[27] Jianmo Ni, Zachary C Lipton, Sharad Vikram, and Julian McAuley, 'Estimating reactions and recommending products with generative models of reviews', in *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pp. 783–791, (2017).

[28] Weike Pan and Li Chen, 'Gbpr: Group preference based bayesian personalized ranking for one-class collaborative filtering.', in *IJCAI*, volume 13, pp. 2691–2697, (2013).

[29] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme, 'Bpr: Bayesian personalized ranking from implicit feedback', in *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence*, pp. 452–461. AUAI Press, (2009).

[30] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller, 'Deterministic policy gradient algorithms', in *ICML*, (2014).

[31] Ilya Sutskever, James Martens, and Geoffrey E Hinton, 'Generating text with recurrent neural networks', in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pp. 1017–1024, (2011).

[32] George Trigeorgis, Konstantinos Bousmalis, Stefanos Zafeiriou, and Björn W Schuller, 'A deep matrix factorization method for learning attribute representations', *IEEE transactions on pattern analysis and machine intelligence*, **39**(3), 417–429, (2017).

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin, 'Attention is all you need', in *Advances in Neural Information Processing Systems*, pp. 5998–6008, (2017).

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

[35] Yongzhen Wang, Xiaozhong Liu, and Zheng Gao, 'Neural related work summarization with a joint context-driven attention mechanism', *arXiv preprint arXiv:1901.09492*, (2019).

[36] Zhongqing Wang and Yue Zhang, 'Opinion recommendation using a neural model', in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1626–1637, (2017).

[37] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, and Alexander J Smola, 'Joint training of ratings and reviews with recurrent recommender networks', (2016).

[38] Chao-Yuan Wu, Amr Ahmed, Alex Beutel, Alexander J Smola, and How Jing, 'Recurrent recommender networks', in *Proceedings of the tenth ACM international conference on web search and data mining*, pp. 495–503. ACM, (2017).

[39] Ziang Xie, 'Neural text generation: A practical guide', *arXiv preprint arXiv:1711.09534*, (2017).

[40] Hong-Jian Xue, Xinyu Dai, Jianbing Zhang, Shujian Huang, and Jiajun Chen, 'Deep matrix factorization models for recommender systems.', in *IJCAI*, pp. 3203–3209, (2017).

[41] Rui Yan, Liang Kong, Congrui Huang, Xiaojun Wan, Xiaoming Li, and Yan Zhang, 'Timeline generation through evolutionary trans-temporal summarization', in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pp. 433–443. Association for Computational Linguistics, (2011).

[42] Hongyu Zang and Xiaojun Wan, 'Towards automatic generation of product reviews from aspect-sentiment scores', in *Proceedings of the 10th International Conference on Natural Language Generation*, pp. 168–177, (2017).

[43] Yongfeng Zhang, 'Explainable recommendation: Theory and applications', *arXiv preprint arXiv:1708.06409*, (2017).

[44] Yongfeng Zhang, Min Zhang, Yiqun Liu, Shaoping Ma, and Shi Feng, 'Localized matrix factorization for recommendation based on matrix block diagonal forms', in *Proceedings of the 22nd international conference on World Wide Web*, pp. 1511–1520. ACM, (2013).