UCEpic: Unifying Aspect Planning and Lexical Constraints for Generating Explanations in Recommendation

Jiacheng Li*
University of California, San Diego
California, USA
j9li@eng.ucsd.edu

Zhankui He*
University of California, San Diego
California, USA
zhh004@eng.ucsd.edu

Jingbo Shang
University of California, San Diego
California, USA
jshang@ucsd.edu

Julian McAuley
University of California, San Diego
California, USA
jmcauley@ucsd.edu

ABSTRACT

Personalized natural language generation for explainable recommendations plays a key role in justifying why a recommendation might match a user's interests. Existing models usually control the generation process by aspect planning. While promising, these aspect-planning methods struggle to generate specific information correctly, which prevents generated explanations from being convincing. In this paper, we claim that introducing lexical constraints can alleviate the above issues. We propose a model, UCEpic, that generates high-quality personalized explanations for recommendation results by unifying aspect planning and lexical constraints in an insertion-based generation manner.

Methodologically, to ensure text generation quality and robustness to various lexical constraints, we pre-train a non-personalized text generator via our proposed robust insertion process. Then, to obtain personalized explanations under this framework of insertion-based generation, we design a method of incorporating aspect planning and personalized references into the insertion process. Hence, UCEpic unifies aspect planning and lexical constraints into one framework and generates explanations for recommendations under different settings. Compared to previous recommendation explanation generators controlled by only aspects, UCEpic incorporates specific information from keyphrases and then largely improves the diversity and informativeness of generated explanations for recommendations on datasets such as RateBeer and Yelp.

CCS CONCEPTS

• **Information systems** → **Recommender systems**; • **Computing methodologies** → **Natural language generation**.

KEYWORDS

Recommender Systems, Explainable Recommendation, Natural Language Generation, Lexical Constraints

*Both authors contributed equally to this research.

KDD '23, August 6–10, 2023, Long Beach, CA, USA

ACM ISBN 979-8-4007-0103-0/23/08.
https://doi.org/10.1145/3580305.3599535

ACM Reference Format:
Jiacheng Li, Zhankui He, Jingbo Shang, and Julian McAuley. 2023. UCEpic: Unifying Aspect Planning and Lexical Constraints for Generating Explanations in Recommendation. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '23), August 6–10, 2023, Long Beach, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3580305.3599535

Table 1: Comparison of previous explanation generators for recommendation in group (A), general lexically constrained generators in group (B), and our UCEpic in group (C).

Group	Methods	Personalized generation	Aspect planning	Lexical constraints	Random keyphrases
(A)	ExpansionNet [34]	✓	✓	✗	✗
	Ref2Seq [32]	✓	✓	✗	✗
	PETER [22]	✓	✓	✗	✗
(B)	NMSTG [40]	✗	✗	✓	✗
	POINTER [45]	✗	✗	✓	✗
	CBART [10]	✗	✗	✓	✓
(C)	Ours	✓	✓	✓	✓

1 INTRODUCTION

Explaining, or justifying, recommendations in natural language is fast gaining traction in recent years [21–24, 28, 32, 34], in order to show product information in a personalized style, and justify how the recommendation meets users' need. That is, given the pair of user and item, the system would generate an explanation such as "*nice TV with 4K display and Dolby Atmos!*". To generate such high-quality personalized explanations which are coherent, relevant and informative, recent studies introduce aspect planning, i.e., including different aspects [22, 23, 32, 34] in the generation process so that the generated explanations will cover those aspects and thus be more relevant to products and to users' interests.

While promising, existing methods struggle to include accurate and highly specific information into explanations because aspects (e.g., *screen* for a TV) mostly control the high-level sentiment or semantics of generated text (e.g., "*good screen and audio*!"), but many informative product attributes are too specific to be accurately generated (e.g., "*nice TV with 4K display and Dolby Atmos!*"). Although aspect-planning explanation generators try to harvest expressive and personalized natural-language-based explanations

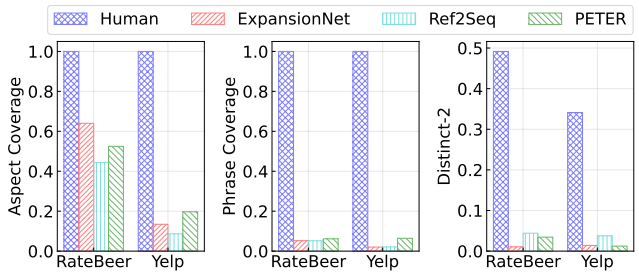

Figure 1: Preliminary experiments on the aspect coverage, phrase coverage, and Distinct-2 of generated explanations from previous models ExpansionNet [34], Ref2Seq [32] and PETER [22] on RateBeer and Yelp datasets. Check details in Appendix A

from users' textual reviews [22, 23, 32, 34], we observe that many informative and specific keyphrases in the training corpus (i.e., user reviews) vanish in generated explanations according to our preliminary experiments. As Figure 1 shows, generated explanations from previous methods miss many specific keyphrases and have much lower Distinct (diversity) scores than a human oracle. Hence, with aspects only, existing methods suffer from generating (1) too general sentences (e.g., "*good screen!*") that are hard to provide diverse and informative explanations to users; (2) sentences with inaccurate details (e.g., "*2K screen*" for a 4K TV), which are not relevant to the product and hurt users' trust.

To address the above problems, we propose to use more concrete constraints to recommendation explanations besides aspects. Specifically, we seek a model unifying *lexical constraints* and *aspect planning*. In this model, introducing lexical constraints guarantees the use of given keyphrases (e.g., "*Dolby Atmos*") and thus includes specific and accurate information. Also, similar to the aspect selection of previous explanation generators [22, 32], such lexical constraints can come from multiple parties. For instance, *explanation systems* select item attributes with some strategies; *vendors* highlight product features; *users* manipulate generated explanations by changing the lexical constraints of interest. Hence, the informativeness, relevance and diversity of generated explanations can be significantly improved compared to previous methods with aspect planning. Meanwhile, aspect planning remains useful when no specific given information but multiple aspects need to be covered.

To achieve this goal of Unifying aspect-planning and lexical Constraints for generating Explanations in Recommendation, we present UCEpic. There are some challenges of building UCEpic. First, lexical constraints are incompatible with existing explanation generation models (see group (A) in Table 1), because they are mostly based on auto-regressive generation frameworks [14, 17, 19, 20, 31, 34] which cannot be guaranteed to contain lexical constraints in any positions with a "left-to-right" generation strategy. Second, although insertion-based generation models (see group (B) in Table 1) are able to contain lexical constraints in generated sentences naturally, we find personalization or aspects cannot be simply incorporated with the "encoder-decoder" framework for existing insertion-based models. Existing tokens are strong signals for new tokens to be predicted, hence the model tends to generate similar sentences and ignore different references[1] from encoders.

For the first challenge, UCEpic employs an insertion-based generation framework and conducts *robust insertion pre-training* on a bi-directional transformer. During robust pre-training, UCEpic gains the basic ability to generate text and handle various lexical constraints. Specifically, inspired by Masked Language Modeling (MLM) [4], we propose an insertion process that randomly inserts new tokens into sentences progressively so that UCEpic is robust to random lexical constraints. For the second challenge, UCEpic uses *personalized fine-tuning* for personalization and awareness of aspects. To tackle the issue of "ignoring references", we propose to view references as part of inserted tokens for the generator and hence the model learns to insert new tokens relevant to references. For aspect planning, we formulate aspects as a special insertion stage where aspect-related tokens will be first generated as a start for the following generation. Finally, lexical constraints, aspect planning and personalized references are unified in the insertion-based generation framework.

Overall, UCEpic is the first explanation generation model unifying aspect planning and lexical constraints. UCEpic significantly improves *relevance*, *coherence* and *informativeness* of generated explanations compared to existing methods. The main contributions of this paper are summarized as follows:

- We show the limitations of only using aspect planning in existing explanation generation, and propose to introduce lexical constraints for explanation generation.
- We present UCEpic including robust insertion pre-training and personalized fine-tuning to unify aspect planning, lexical constraints and references in an insertion-based generation framework.
- We conduct extensive experiments on two datasets. Objective metrics and human evaluations show that UCEpic can largely improve the diversity, relevance, coherence and informativeness of generated explanations.

2 RELATED WORK

Explanation Generation For Recommendation. Generating explanations of recommended items for users has been studied for a long time with various output formats [6, 43, 44] (e.g., item aspects, attributes, similar users). Recently, natural-language-based explanation generation has drawn great attention [21–24, 28, 32, 34] to generate post-hoc explanations or justifications in personalized style. For example, Li et al. [24] applied RNN-based model to generate explanations based on predicted ratings. To better control the explanation generation process, Ni et al. [32] extracted aspects and controlled the semantics of generated explanations conditioned on different aspects, and Li et al. [22] proposed a personalized transformer model to generate explanations based on given item features. Also, the review generation area is highly related since explanation generation methods usually harvest expressive and informative explanations from user reviews. Many controllable review generators [5, 34, 39] are tailored to explanation generations as baseline

[1] In literature [32], the terminology *references* refer to a user's personalized textual data such as the historical product reviews.

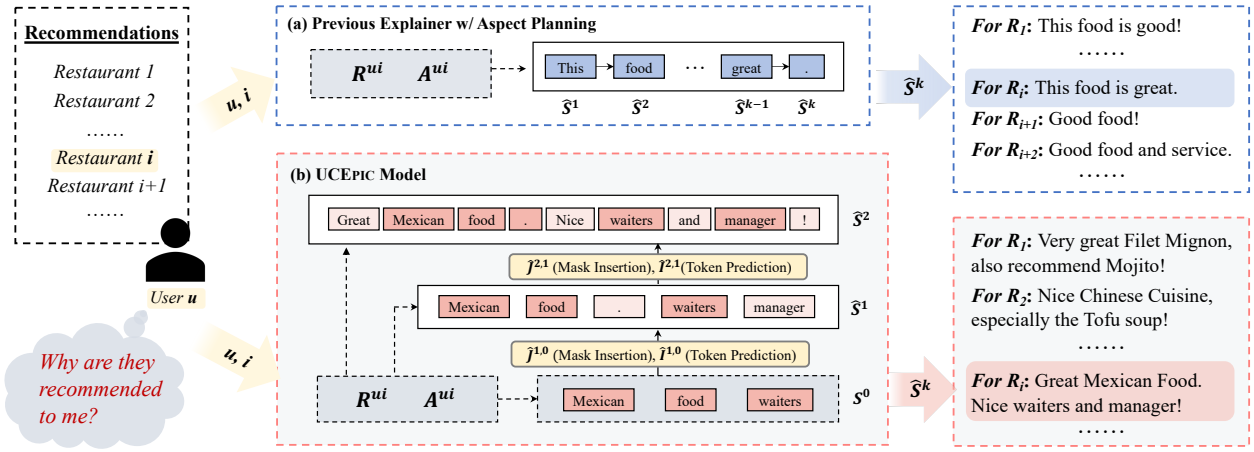

Figure 2: Overview of generating explanations for a given user and recommended items using (a) an aspect-planning autoregressive generation model; using (b) our UCEpic that unifies aspect-planning and lexical constraints.

models in early experiments. Although previous works continued increasing the controllability of generation, they are all on the basis of auto-regressive generation frameworks [14, 17, 19, 20, 31, 34] thus only considering aspect planning. In our work, UCEpic increases the controllability, informativeness of generated explanations by unifying aspect planning and lexical constraints under an insertion-based generation framework.

Lexically Constrained Text Generation. Lexically constrained generation requires that generated text contain the lexical constraints (e.g., keywords). Early works usually involve special decoding methods. Hokamp and Liu [12] proposed a lexical-constrained grid beam search decoding algorithm to incorporate constraints. Post and Vilar [36] presented an algorithm for lexically constrained decoding with reduced complexity in the number of constraints. Hu et al. [13] further improved decoding by a vectorized dynamic beam allocation. Miao et al. [30] introduced a sampling-based conditional decoding method, where the constraints are first placed in a template, then decoded words under a Metropolis-Hastings sampling. Special decoding methods usually need a high running time complexity. Recently, Zhang et al. [45] implemented hard-constrained generation with $O(\log n)$ time complexity by language model pre-training and insertion-based generation [2, 7, 8, 37] used in machine translation. CBART [10] uses the pre-trained model BART [15] and the encoder and decoder are used for instructing insertion and predicting mask respectively.

3 METHODOLOGY

We describe aspect planning and lexical constraints for explanation generation as follows. Given a user persona R^u, item profile R^i for user u and item i as references, the generation model under aspect planning outputs the explanation E^{ui} related to an aspect A^{ui} but not necessarily including some specific words. Whereas for lexical constraints, given several lexical constraints (e.g, phrases or keywords) $\mathrm{C}^{ui} = \{c_1, c_2, \ldots, c_m\}$, the model will generate an explanation $\mathrm{E}^{ui} = (w_1, w_2, \ldots, w_n)$ that has to exactly include all given lexical constraints c_i, which means $c_i = (w_j, \ldots, w_k)$. The lexical constraints can be from users, businesses, or item attributes recommended by personalized systems in a real application. UCEpic unifies the two kinds of constraints in one model [2]. We study only the explanation generation method and assume aspects and lexical constraints are given. Our notations are summarized in Table 2

Table 2: Notation.

Notation	Description
R^u, R^i	historical review profile of user u and item i.
E^{ui}	generated explanation when item i is recommended to user u.
A^{ui}	aspects controlling explanation generation for item i and user u.
C^{ui}	lexical constraints (e.g., keywords) controlling explanation generation for item i and user u.
S^k, $\hat{S}^k$	text sequence of the k-th stage generation. S^k is training data and $\hat{S}^k$ is model prediction.
$I^{k,k-1}$, $\hat{I}^{k,k-1}$	intermediate sequence between S^{k-1} and S^k. (training data and model prediction)
$J^{k,k-1}$, $\hat{J}^{k,k-1}$	insertion number sequence between S^{k-1} and S^k. (training data and model prediction)
$\mathbf{D}$	a bi-directional transformer for encoding.
$\mathbf{H}_{MI}$	a linear projection layer for insertion numbers.
$\mathbf{H}_{TP}$	a multilayer perceptron with activation function for token prediction.

3.1 Robust Insertion

3.1.1 Motivation. Previous explanation generation methods [22, 32] generally adopt auto-regressive generation conditioned on some personalized inputs (e.g., personalized references and aspects). As shown in Figure 2 (a), the auto-regressive process generates words in a "left-to-right" direction so lexical constraints are difficult to be contained in the generation process. However, for the insertion-based generation in Figure 2 (b) which progressively inserts new

[2] UCEpic has two modes: generating under aspect planning or generating under lexical constraints

Table 3: Data construction examples.

Data	Example
S^K (sentence)	<s>Good tacos. Love the crispy citrus + tropical fruits flavor. </s>
$I^{K,K-1}$	<s>[MASK] tacos. Love the [MASK] [MASK] + tropical fruits flavor. </s>
$J^{K,K-1}$	[1 0 0 0 2 0 0 0 0 0 0]
S^{K-1}	<s>tacos. Love the + tropical fruits flavor. </s>
...	...
S^0 (lexical constraints)	<s>tropical fruits flavor </s>

tokens based on existing words, lexical constraints can be easily contained by viewing constraints as a starting stage of insertion.

3.1.2 Formulation. The insertion-based generation can be formulated as a progressive sequence of K stages $S = \{S^0, S^1, \ldots, S^{K-1}, S^K\}$, where S^0 is the stage of lexical constraints and S^K is our final generated text. For $k \in \{1, \ldots, K\}$, S^{k-1} is a sub-sequence of S^k. The generation procedure finishes when UCEpic does not insert any new tokens into S^K. In the training process, all sentences are prepared as training pairs. Specifically, pairs of text sequences are constructed at adjacent stages (S^{k-1}, S^k) that reverse the insertion-based generation process. Each explanation E^{ui} in the training data is broken into a consecutive series of pairs: $(S^0, S^1), (S^1, S^2), \ldots, (S^{K-1}, S^K)$, and when we construct the training data, the final stage S^K is our explanation text E^{ui}.

3.1.3 Data Construction. Given a sequence stage S^k, we obtain the previous stage S^{k-1} by two operations, masking and deletion. Specifically, we randomly mask the tokens in a sequence by probability p as MLM to get the intermediate sequence $I^{k,k-1}$. Then, [MASK] tokens are deleted from the intermediate sequence $I^{k,k-1}$ to obtain the stage S^{k-1}. The numbers of deleted [MASK] tokens after each token in $I^{k,k-1}$ are recorded as an insertion number sequence $J^{k,k-1}$. Finally, each training instance contains four sequences $(S^{k-1}, I^{k,k-1}, J^{k,k-1}, S^k)$. We include a simple example for the data construction process in Table 3. Since we delete $T * \mathrm{p}$ tokens in sequence S^k where T is the length of S^k, the average number of K is $\log_{\frac{1}{1-\mathrm{p}}} T$. Models trained on this data will easily re-use the knowledge from BERT-like models which have a similar pre-training process of masked word prediction.

Insertion generation (see Algorithm 1) is an inverse process of data construction. For each insertion step prediction from $\hat{S}^{k-1}$ to $\hat{S}^k$, the model will recover text sequences by two operations, mask insertion and token prediction. In particular, UCEpic first inserts [MASK] tokens between any two existing tokens in $\hat{S}^{k-1}$ to get $\hat{I}^{k,k-1}$ according to $\hat{J}^{k,k-1}$ predicted by an insertion prediction head. Then, UCEpic with a language modeling head predicts the masked tokens in $\hat{I}^{k,k-1}$, and recovers [MASK] tokens into words to obtain the $\hat{S}^k$.

3.1.4 Modules. UCEpic uses a bi-directional Transformer architecture with two different prediction heads for mask insertion and token prediction. The architecture of the model is closely related to that used in RoBERTa [26]. The bi-directional Transformer $\mathbf{D}$ will predict the mask insertion numbers and word tokens with two heads $\mathbf{H}_{MI}$ and $\mathbf{H}_{TP}$ respectively. $\mathbf{H}_{TP}$ is a multilayer perceptron (MLP) with activation function GeLU [11] and $\mathbf{H}_{MI}$ is a linear projection layer. Finally, our predictions of mask insertion numbers and word tokens are computed as:

Algorithm 1 Insertion in the k-th Stage

procedure INSERTION($\hat{S}^{k-1}$)
  $\hat{J}^{k,k-1} \leftarrow$ predict number of masks from $\hat{S}^{k-1}$ via eq. (1) ;
  $\hat{I}^{k,k-1} \leftarrow$ build intermediate sequence from $\hat{J}^{k,k-1}$ and $\hat{S}^{k-1}$;
  $\hat{S}^k \leftarrow$ predict masked tokens in $\hat{I}^{k,k-1}$ via eq. (2);
  return predicted sequence $\hat{S}^k$;

$$y_{MI} = \mathbf{H}_{MI}(\mathbf{D}(\hat{S}^{k-1})),\ \hat{J}^{k,k-1} = \mathrm{argmax}(y_{MI}) \tag{1}$$

$$y_{TP} = \mathbf{H}_{TP}(\mathbf{D}(\hat{I}^{k,k-1})),\ \hat{S}^k = \mathrm{argmax}(y_{TP}) \tag{2}$$

where $y_{MI} \in \mathbb{R}^{l_s \times d_{ins}}$ and $y_{TP} \in \mathbb{R}^{l_I \times d_{vocab}}$, l_s and l_I are the length of $\hat{S}^{k-1}$ and $\hat{I}^{k,k-1}$ respectively, d_{ins} is the maximum number of insertions and d_{vocab} is the size of vocabulary. $\hat{I}^{k,k-1}$ is obtained by inserting [MASK] tokens into $\hat{S}^{k-1}$ according to $\hat{J}^{k,k-1}$.

Because the random insertion process is more complicated to learn than the traditional autoregressive generation process, we first pre-train UCEpic with our robust insertion method for general text generation without personalization. The pre-trained model can generate sentences from randomly given lexical constraints.

3.2 Personalized References and Aspect Planning

3.2.1 Motivation. To incorporate personalized references and aspects, one direct method is to have another text and aspect encoder and insertion generation conditioned on the encoder like the sequence-to-sequence model [38]. However, we find the pre-trained insertion model with another encoder will generate similar sentences with different personalized references and aspects. The reason is the pre-trained insertion model views the lexical constraints or existing tokens in text sequences as a strong signal to determine new inserted tokens. Even if our encoder provides personalized features, the model tends to overfit features from existing tokens. Without lexical tokens providing different starting stages, generated sentences are usually the same.

3.2.2 Formulation. To better learn personalization, we propose to view references and aspects as special existing tokens during the insertion process. Specifically, we construct a training stage S_+^k to include references and aspects as:

$$\begin{aligned} S_+^k &= [\mathrm{R}^{ui}, \mathrm{A}^{ui}, S^k] \\ &= [w_0^r, \ldots, w_{|\mathrm{R}^{ui}|}^r, w_0^a, \ldots, w_{|\mathrm{A}^{ui}|}^a, w_0, \ldots, w_{|S^k|}] \end{aligned} \tag{3}$$

where R^{ui}, A^{ui} denote personalized references and aspects; w^r, w^a and w are tokens or aspect ids in references, aspects and insertion stage tokens respectively. Because insertion-based generation relies on token positions to insert new tokens, we create token position ids in Transformer starting from 0 for R^{ui}, A^{ui} and S^k respectively in order to make it consistent for S^k between pre-training and fine-tuning. Similarly, we obtain the insertion number sequence $J_+^{k,k-1} = [\mathbf{0}_{|\mathrm{R}^{ui}|}, \mathbf{0}_{|\mathrm{A}^{ui}|}, J^{k,k-1}]$ and intermediate training stage $I_+^{k,k-1} = [\mathrm{R}^{ui}, \mathrm{A}^{ui}, I^{k,k-1}]$, where $\mathbf{0}_{|\mathrm{R}^{ui}|}$ and $\mathbf{0}_{|\mathrm{A}^{ui}|}$ are zero vectors

which have same length as R^{ui} and A^{ui} respectively, because we do not insertion any tokens into references and aspects.

3.2.3 Modules. We encode $\hat{S}_+^k$ and $\hat{I}_+^{k,k-1}$ with bi-directional Transformer $\mathbf{D}$ to get the insertion numbers y_{MI} and predicted tokens y_{TP} as follows:

$$[\mathrm{O}_S^{R^{ui}}, \mathrm{O}_S^{A^{ui}}, \mathrm{O}^{S^k}] = \mathbf{D}(\hat{S}_+^k) \tag{4}$$

$$[\mathrm{O}_I^{R^{ui}}, \mathrm{O}_I^{A^{ui}}, \mathrm{O}^{I^{k,k-1}}] = \mathbf{D}(\hat{I}_+^{k,k-1}) \tag{5}$$

$$y_{MI} = \mathbf{H}_{MI}(\mathrm{O}^{S^k}) \tag{6}$$

$$y_{TP} = \mathbf{H}_{TP}(\mathrm{O}^{I^{k,k-1}}) \tag{7}$$

Similar as Equation (1) and Equation (2), we can get $\hat{J}^{k,k-1}$ and $\hat{S}^k$ by argmax operation. Because personalized references and aspects are viewed as special existing tokens, UCEpic will directly incorporate token-level information as generation conditions and hence generates diverse explanations.

Recall that existing text sequences are strong signals for token prediction. For better aspect-planning generation, we design two starting stages S_{+a}^0 and S_{+l}^0 for aspects and lexical constraints respectively. In particular, we expect the aspect-related tokens can be generated at the starting stage (i.e., no existing tokens) according to given aspects and personalized references. Hence, the aspect starting stage is $S_{+a}^0 = [\mathrm{R}^{ui}, \mathrm{A}^{ui}]$. Lexical constraint starting stage is $S_{+l}^0 = [\mathrm{R}^{ui}, \mathrm{A}^{pad}, \mathrm{C}^{ui}]$ where A^{pad} is a special aspect that is used for lexical constraints. During training, we sample S_{+a}^0 with probability p to ensure UCEpic learns aspect-related generation effectively which is absent in pre-training.

3.3 Model Training

The training process of UCEpic is to learn the inverse process of data generation. Given stage pairs (S_+^{k-1}, S_+^k) and training instance $(S_+^{k-1}, I_+^{k,k-1}, J_+^{k,k-1}, S_+^k)$ from pre-processing [3], we optimize the following objective:

$$\begin{aligned}
\mathcal{L} &= -\log p(S_+^k | S_+^{k-1}) \\
&= -\log \underbrace{p(S_+^k, J_+^{k,k-1} | S_+^{k-1})}_{\text{Unique } J \text{ assumption}} \\
&= -\log p(S_+^k | J_+^{k,k-1}, S_+^{k-1}) p(J_+^{k,k-1} | S_+^{k-1}) \\
&= -\log \underbrace{p(S_+^k | I_+^{k,k-1})}_{\text{Token prediction}} \underbrace{p(J_+^{k,k-1} | S_+^{k-1})}_{\text{Mask insertion}}, \\
&\text{where } I_+^{k,k-1} = \text{MaskInsert}(J_+^{k,k-1}, S_+^{k-1})
\end{aligned} \tag{8}$$

where MaskInsert denotes the mask token insertion. We make a reasonable assumption that $J_+^{k,k-1}$ *is unique given* (S_+^k, S_+^{k-1}). This assumption is usually true unless in some corner cases multiple $J_+^{k,k-1}$ could be legal (e.g., masking one "moving" word in "a moving moving moving van"); $I_+^{k,k-1}$ by definition is the intermediate sequence, which is equivalent to the given $(J_+^{k,k-1}, S_+^{k-1})$. In Equation (8), we jointly learn (1) likelihood of mask insertion number for each token from UCEpic with $\mathbf{H}_{MI}$, and (2) likelihood of word tokens for the masked tokens from UCEpic with $\mathbf{H}_{TP}$.

Similar to training BERT [4], we optimize only the masked tokens in token prediction. The selected tokens to mask have the probability 0.1 to stay unchanged and the probability 0.1 to be randomly replaced by another token in the vocabulary. For mask insertion number prediction, most numbers in $J_+^{k,k-1}$ are 0 because we do not insert any tokens between the existing two tokens in most cases. To balance the insertion number, we randomly mask the 0 in $J_+^{k,k-1}$ with probability q. Because our mask prediction task is similar to masked language models, the pre-trained weights from RoBERTa [26] can be naturally used for initialization of UCEpic to obtain prior knowledge.

Table 4: Statistics of datasets

Dataset	Train	Dev	Test	#Users	#Items	#Aspects
RateBeer	16,839	1,473	912	4,385	6,183	8
Yelp	252,087	37,662	12,426	235,794	22,412	59

3.4 Inference

At inference time, we start from the given aspects A^{ui} or lexical constraint C^{ui} to construct starting stage S_{+a}^0 or S_{+l}^0 respectively. Then, UCEpic predicts $\{\hat{S}_+^1, \ldots, \hat{S}_+^K\}$ repeatedly until no additional tokens generated or reaching the maximum stage number. We obtain final generated explanation $\hat{S}^K$ from $\hat{S}_+^K$ by removing R^{ui} and A^{ui}. Without loss of generality, we show the inference details from $\hat{S}_+^{k-1}$ stage to $\hat{S}_+^k$ stage: (1) given $\hat{S}_+^{k-1}$ UCEpic uses $\mathbf{H}_{MI}$ to predict $\hat{J}_+^{k,k-1}$ insertion number sequence[4]; (2) given $\hat{I}_+^{k,k-1}$ from MaskInsert$(\hat{J}_+^{k,k-1}, \hat{S}_+^{k-1})$, UCEpic can use $\mathbf{H}_{TP}$ to predict $\hat{S}_+^k$ with a specific decoding strategy (e.g., greedy search, top-K sampling). (3) given $\hat{S}_+^k$, UCEpic meets the termination requirements or executes step (1) again. The termination criterion can be a maximum interation number or UCEpic does not insert new tokens into $\hat{S}_+^k$.

4 EXPERIMENTS

4.1 Datasets

For *pre-training*, we use English Wikipedia[5] for robust insertion training which has 11.6 million sentences. For fair comparison with baselines pre-trained on a general corpus, we use Wikipedia as the pre-training dataset; and for *fine-tuning*, we use Yelp[6] and RateBeer [29] to evaluate our model (see Table 4). We further filter the reviews with a length larger than 64. For each user, following Ni et al. [32], we randomly hold out two samples from all of their reviews to construct the development and test sets. Following previous works [32, 33], we employ an unsupervised aspect extraction tool [18] to obtain phrases and corresponding aspects for lexical constraints and aspect planning respectively. The number of aspects for each dataset is determined by the tool automatically and aspects provide coarse-grained semantics of generated explanations.

[3] For fine-tuning with personalized references and aspects, we train the model with stage pairs (S_+^{k-1}, S_+^k) and training instance $(S_+^{k-1}, I_+^{k,k-1}, J_+^{k,k-1}, S_+^k)$

[4] We set predicted insertion number as 0 for given phrases in S_{+l}^0, to prevent given phrases from modification.

[5] https://dumps.wikimedia.org/

[6] https://www.yelp.com/dataset

Table 5: Performance comparison of the explanation generation models (ExpansionNet, Ref2Seq, PETER), lexically constrained generation models (NMSTG, POINTER, CBART) and UCEpic. All values are in percentage (%). We underline the highest scores of aspect-planning generation results and the highest scores of lexically constrained generation are bold.

Models	RateBeer							Yelp						
	B-1	B-2	D-1	D-2	M	R	BS	B-1	B-2	D-1	D-2	M	R	BS
Human-Oracle	–	–	8.30	49.16	–	–	–	–	–	3.8	34.1	–	–	–
Aspect-planning generation														
ExpansionNet	8.96	1.79	0.20	1.05	16.30	10.13	75.58	4.92	0.47	0.18	1.40	7.78	5.42	76.27
Ref2Seq	17.15	4.17	0.95	4.41	16.66	15.66	80.76	8.34	0.98	0.46	3.77	7.58	11.19	82.66
PETER	25.25	5.35	0.74	3.44	19.19	20.34	84.03	14.26	2.25	0.26	1.23	12.25	14.75	82.55
UCEpic	27.42	2.89	4.49	29.23	19.54	15.48	83.53	8.03	0.72	1.89	14.75	8.10	11.58	83.53
Lexically constrained generation														
ExpansionNet	5.41	0.49	0.97	4.91	6.09	5.55	76.14	1.49	0.08	0.40	1.90	2.19	1.93	73.68
Ref2Seq	17.94	4.50	1.09	5.49	17.03	15.17	83.72	6.38	0.77	0.51	3.64	7.02	10.58	82.88
PETER	15.03	2.46	2.04	11.40	9.49	13.27	79.08	7.59	1.32	1.52	8.70	7.64	12.24	80.89
NMSTG	22.82	2.30	6.02	50.39	15.17	15.35	82.31	13.67	0.77	**4.57**	**57.02**	9.64	11.13	80.80
POINTER	6.00	0.31	**11.24**	**56.02**	7.41	11.21	81.80	1.50	0.06	5.49	29.76	3.24	5.23	80.85
CBART	2.49	0.54	8.49	34.74	8.45	13.84	83.30	2.19	0.60	5.32	26.79	9.41	15.00	84.08
UCEpic	**27.97**	**5.09**	5.24	32.04	**19.90**	**17.05**	**84.03**	**13.77**	**3.06**	2.85	20.39	**14.45**	**16.92**	**84.55**

Note that, typically, the number of aspects is much smaller than the number of lexical constraints and aspects are more high-level.

4.2 Baselines

We consider two groups of baselines for automatic evaluation to evaluate model effectiveness. The first group is existing text generation models for recommendation with *aspect planning*.

- **ExpansionNet** [34], generates reviews conditioned on different aspects extracted from a given review title or summary.
- **Ref2Seq** [32], a Seq2Seq model incorporates contextual information from reviews and uses fine-grained aspects to control explanation generation.
- **PETER** [22], a Transformer-based model that uses user- and item-IDs and given phrases to predict the words in target explanation generation. This baseline can be considered as a state-of-the-art model for explainable recommendation.

We compare those baselines under both aspect planning and lexical constraints. Specifically, we feed lexical constraints (i.e., keyphrases) into models and expect models copy keyphrases to generated text.

The second group includes general natural language generation models with *lexical constraints*:

- **NMSTG** [40], a tree-based text generation scheme that from given lexical constraints in prefix tree form, the model generates words to its left and right, yielding a binary tree.
- **POINTER** [45], an insertion-based generation method pretrained on constructed data based on dynamic programming.
- **CBART** [10], uses the pre-trained BART [15] and instructs the decoder to insert and replace tokens by the encoder.

The second group of baselines cannot incorporate aspects or personalized information as references. These models are trained and generate text solely based on given lexical constraints. We do not include explanation generation methods such as NRT [3], Att2Seq [5] and ReXPlug [9], non-natural-language explainable recommenders such as EFM [44] and DEAML [6], and lexically constrained methods CGMH [30], GBS [12] because PETER and CBART reported better performance than these models. We also had experiments with "encoder-decoder" based UCEpic as mentioned in Section 1, but this model generates same sentences for all user-item pairs hence we do not include it as a baseline. Detailed settings of baselines can be found in Appendix B.

4.3 Evaluation Metrics

We evaluate the generated sentences from two aspects: generation quality and diversity. Following Ni et al. [32], Zhang et al. [45], we use n-gram metrics including BLEU (B-1 and B-2) [35], METEOR (M) [1] and ROUGE-L (R-L) [25] which measure the similarity between the generated text and human oracle. As for generation diversity, we use Distinct (D-1 and D-2) [16]. We also introduce BERT-score (BS) [42] as a semantic rather than n-gram metrics.

4.4 Implementation Details

We use `RoBERT-base` [26] (#params $\approx$ 130M, other pretrained model sizes are listed in Appendix B). In training data construction, we randomly mask p = 0.2 tokens in S^k to obtain $I^{k,k-1}$. 0 in $J^{k,k-1}$ are masked by probability q = 0.9. The tokenizer is byte-level BPE following RoBERTa. For *pre-training*, the learning rate is 5e-5, batch size is 512 and our model is optimized by AdamW [27] in 1 epoch. For *fine-tuning* on downstream tasks, the learning rate is 3e-5, and the batch size is 128 with the same optimizer as pre-training. The training epoch is 10 and we select the best model on the development set as our final model which is evaluated on test data. We randomly sample one aspect and one phrase from

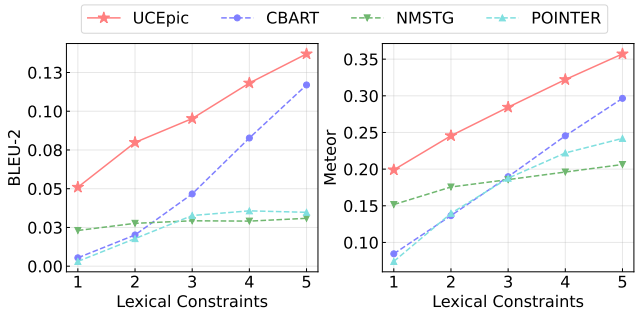

Figure 3: Performance (i.e., B-2 and Meteor) of lexically constrained generation models on RateBeer data with different numbers of keyphrases.

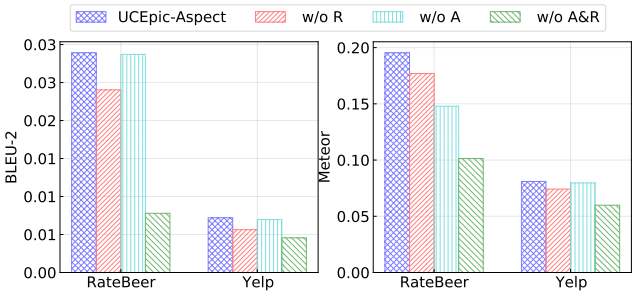

Figure 4: Ablation study on aspects and references.

the target text as the aspect for planning and lexical constraint respectively [7].

4.5 Automatic Evaluation

4.5.1 Overall Performance. In Table 5, we report evaluation results for different generation methods. For aspect-planning generation, UCEpic can achieve comparable results as the state-of-the-art model PETER. Specifically, although PETER obtains better B-2 and ROUGE-L than our model, the results from UCEpic are significantly more diverse than PETER. A possible reason is that auto-regressive generation models such as PETER tend to generate text with higher n-gram metric results than insertion-based generation models, because auto-regressive models generate a new token solely based on left tokens while (insertion-based) UCEpic considers tokens in both directions. Despite the intrinsic difference, UCEpic still achieves comparable B-1, Meteor and BERT scores with PETER.

Under the lexical constraints, the results of existing explanation generation models become lower than the results of aspect-planning generation which indicates current explanation generation models struggle to include specific information (i.e., keyphrases) in explanations. Although current lexically constrained generation methods produce text with high diversity, they tend to insert less-related tokens with users and items. Hence, the generated text is less coherent (low n-gram metric results) than UCEpic because these methods cannot incorporate user personas and item profiles from references which are important for explainable recommendation. In contrast, UCEpic easily includes keyphrases in explanations and learns user-item information from references. Therefore, our model largely outperforms existing explanation generation models and lexically constrained generation models.

Based on the discussion, we argue UCEpic unifies the aspect planning and lexical constraints for explainable recommendations.

4.5.2 Number of Lexical Constraints. Figure 3 shows the performance of lexically constrained generation models under different keyphrase numbers. Overall, UCEpic consistently outperforms other models under different numbers of lexical constraints. In particular, NMSTG and POINTER do not achieve a large improvement as the number of keyphrases increases because they cannot have random keywords and given phrases are usually broken into words. The gap between UCEpic and CBART becomes large as the number of keyphrases decreases since CBART cannot obtain enough information for explanation generation with only a few keywords, but UCEpic improves this problem by incorporating user persona and item profiles from references. The results indicate existing lexically constrained generation models cannot be applied for explanation generation with lexical constraints.

[7] More details are in https://github.com/JiachengLi1995/UCEpic. We also released an additional checkpoint pre-trained on personalized review datasets including Amazon Reviews [32] and Google Local [41].

Table 6: UCEpic with different constraints on Yelp dataset. L denotes lexical constraints.

Constraints	B-1	B-2	D-1	D-2	M	R	BS
Aspect	8.03	0.72	1.89	14.75	8.09	11.58	83.53
L-Extract	13.77	3.06	2.85	20.39	14.45	16.92	84.55
L-Frequent	10.05	0.87	2.02	15.88	9.14	12.23	83.73
L-Random	9.81	0.79	3.00	21.04	8.73	11.61	83.50
Aspect & L	13.12	3.01	2.89	20.34	14.41	16.94	84.56

4.5.3 Ablation Study. To validate the effectiveness of our unifying method and the necessity of aspects and references for explanation generation, we conduct an ablation study on two datasets and the results are shown in Figure 4. We train our model and generate explanations without aspects (w/o A), without references (w/o R) and without both of them (w/o A&R). From the results, we can see that BLEU-2 and Meteor decrease if we do not give aspects to the model because the aspects can guide the semantics of explanations. Without references, the model generates similar sentences which usually contain high-frequency words from the training data. The performance drops markedly if both references and aspects are absent from the model. Therefore, our unifying method for references and aspects is effective and provides user-item information for explanation generation.

4.5.4 Kind of Constraints. We study the performance of UCEpic with different kinds of constraints on the Yelp dataset and the results are shown in Table 6. The settings of Aspect and L-Extract are consistent as UCEpic under aspect-planning and lexical constraints [8] respectively in Table 5. We also study three other kinds of constraints: (1) L-Frequent. We use the most frequent noun phrase of an item as the lexical constraint. (2) L-Random. We randomly sample the lexical constraint from all noun phrases of an item.

[8] Aspects and phrases are extracted from the generation target and we randomly sampled one aspect and one phrase as model inputs.

Table 7: Generated explanations from Yelp dataset. Lexical constraints (phrases) are highlighted in explanations.

Phrases	**pepper chicken**	**north shore , meat**
Human	Food was great. The pepper chicken is the best. This place is neat and clean. The staff are sweet. I recomend them to anyone!!	Great Italian food on the north shore ! Menu changes daily based on the ingredients they can get locally. Everything is organic and made "clean". There is no freezer on the property, so you know the meat was caught or prepared that day. The chef is also from Italy! I highly recommend!
Ref2Seq	best restaurant in town ! ! !	what a good place to eat in the middle of the area . the food was good and the service was good .
PETER	This place is great! I love the food and the service is always great. I love the chicken and the chicken fried rice. I love this place.	The food was good, but the service was terrible. The kitchen was not very busy and the kitchen was not busy. The kitchen was very busy and the kitchen was not busy.
POINTER	pepper sauce chicken !	one of the best restaurants in the north as far as i love the south shore . great meat ! !
CBART	Great spicy pepper buffalo wings and chicken wings.	Best pizza on the north shore ever! Meatloaf is to die for, especially with meat lovers.
UCEpic	Great Chinese restaurant, really great food! The customer service are amazing! Everything is delicious and delicious! I think this local red hot pepper chicken is the best.	I had the best Italian north shore food. The service is great, meat that is fresh and delicious. Highly recommend!

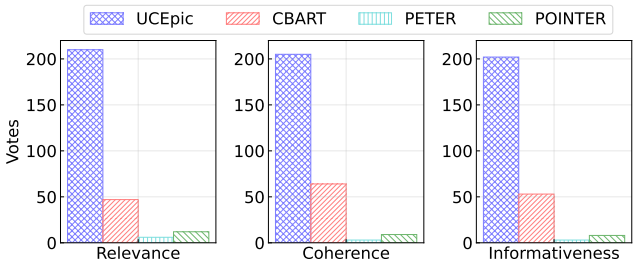

Figure 5: Human evaluation on explanation quality.

(3) Aspect & L. This method combines both aspect-planning and lexical constraints demonstrated in Table 5 and uses the two kinds of constraints simultaneously. From the results, we can see that (1) L-Extract and Aspect & L have similar results which indicate the lexical constraints have strong restrictions on the generation process hence the aspect planning rarely has controllability on the results. (2) Generation with lexical constraints can achieve better results than aspect-planning generation. (3) Lexical constraint selections (i.e., L-Extract, L-Frequent, L-Random) result in significant differences in generation performance, which movitate that lexical constraint selections can be further explored in future work.

4.6 Human Evaluation

We conduct a human evaluation on generated explanations. Specifically, We sample 500 ground-truth explanations from Yelp dataset, then collect corresponding generated explanations from PETER-aspect, POINTER, CBART and UCEpic respectively. Given the ground-truth explanation, annotator is requested to select the *best* explanation on different aspects i.e., *relevance*, *coherence* and *informativeness*) among explanations generated from PETER. POINTER, CBART and UCEpic (see Appendix C for details). We define *relevance*, *coherence* and *informativeness* as:

- **Relevance:** the details in the generated explanation are consistent and relevant to the ground-truth explanations.
- **Coherence:** the sentences in the generated explanation are logical and fluent.
- **Informativeness:** the generated explanation contains specific information, instead of vague descriptions only.

The voting results are shown in Figure 5. We can see that UCEpic largely outperforms other methods in all aspects especially for relevance and informativeness. In particular, lexically constrained generation methods (UCEpic and CBART) significantly improve the quality of explanations because specific product information can be included in explanations by lexical constraints. Because POINTER is not robust to random keyphrases, the generated explanations do not get improvements from lexical constraints.

4.7 Case Study

We compare generated explanations from existing explanation generation models (i.e., Ref2Seq, PETER), lexically constrained generation models (i.e., POINTER, CBART) and UCEpic in Table 7. We can see that Ref2Seq and PETER usually generate general sentences which are not informative because they struggle to contain specific item information by traditional auto-regressive generation. POINTER and CBART can include the given phrases (pepper chicken) in their generation, but they are not able to learn information from references and hence generate some inaccurate words (pepper sauce chicken, chicken wings) which mislead users. In contrast, UCEpic can generate coherent and informative explanations which include the specific item attributes and are highly relevant to the recommended item.

5 CONCLUSION

In this paper, we propose to have lexical constraints in explanation generation which can largely improve the informativeness

and diversity of generated reviews by including specific information. To this end, we present UCEpic, an explanation generation model that unifies both aspect planning and lexical constraints in an insertion-based generation framework. We conduct comprehensive experiments on RateBeer and Yelp datasets. Results show that UCEpic significantly outperforms previous explanation generation models and lexically constrained generation models. Human evaluation and a case study indicate UCEpic generates coherent and informative explanations that are highly relevant to the item.

A MOTIVATING EXPERIMENT DETAILS

In this experiment, we evaluate the diversity and informativeness of explanations. Specifically, we apply phrase coverage, aspect coverage and Distinct-2 to measure generated explanations and human-written explanations.

For **phrase coverage**, we extract noun phrases from explanations by spaCy noun chunks. Then we compare the phrases in human-written explanations and generated explanations. If a phrase appears in both explanations, we consider it as a covered phrase by generated explanations. This experiment measures how much specific information can be included in the generated explanations.

For **aspect coverage**, we use the aspect extraction tool [18] per dataset to construct a table that maps phrases to aspects, then we map the phrases in generated explanations to aspects by looking up the phrase-aspect table. For each sample, we calculate how many aspects in ground-truth explanation are covered in generated explanations and report the average aspect coverage per dataset.

For **Distinct-2**, we use the numbers as described in Table 5.

B BASELINE DETAILS

For **ExpansionNet**, we use the default setting which uses hidden size 512 for RNN encoder and decoder, batch size as 25 and learning rate 2e-4. For aspect planning in ExpansionNet, we use the set of lexical constraints (as concatenated phrases) to replace the *title* or *summary* input as contextual information for training and testing.

For **Ref2Seq**, we use the default setting with 256 hidden size, 512 batch size and 2e-4 learning rate. For aspect planning, we concatenate our given phrases as references (historical explanations are also incorporated as references following the original implementation) as contextual information in training and testing.

For **PETER**, we use the original setting with 512 embedding size, 2048 hidden units, 2 self-attention heads with 2 transformer layers, 0.2 dropout. We use the training strategy suggested by the authors. Since original PETER only supports single words as an aspect, we adopt PETER to multiple words with a maximum length of 20 and reproduce the original single-word model on our multi-word model. We input our lexical constraints as the multi-word input for PETER training and testing.

For **NMSTG**, we use the default settings with an LSTM with 1024 hidden size with the uniform oracle. We convert our lexical constraints into a prefix sub-tree as the input of NMSTG, and then use the best sampling strategy in our testing (i.e., `StochasticSampler`) for NMSTG.

For **POINTER**, we use the pre-training `BERT-large` [4] (#params ≈ 340M.) from WIKI to fine-tune 40 epochs on our downstream datasets. We use all the default settings except batch sizes since POINTER requires 16 GPUs for distributed training that exceeds our computational resources. Instead, we train POINTER with the same configuration on 3 GPUs. For testing, we select the base maximum turn as 3 with the default greedy decoding strategy. We feed lexical constraints as the original implementation.

For **CBART**, we use the checkpoint pre-trained on `BERT-large` [4] (#params ≈ 340M.) with the one-billion-words dataset to fine-tune our downstream datasets. We use the 'tf-idf' training mode and finetune it on one GPU. For testing, we select the greedy decoding strategy. We set other hyper-parameters to default as the code base [9].

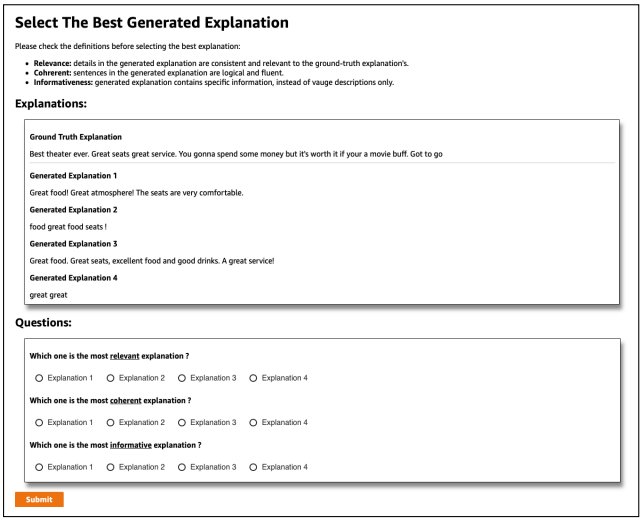

Figure 6: Our human evaluation example on MTurk.

C HUMAN EVALUATION DETAILS

We conduct human evaluation experiments on Yelp datasets to evaluate the generation quality of generated explanations in terms of *relevance*, *coherence* and *informativeness*.

We submit our task to MTurk [10] and set the reward as $0.02 per question. For each question, we first show the definition of *relevance*, *coherence* and *informativeness*, then we shuffle the order of model-generated explanations to eliminate the positional bias. Each question is requested to be answered by 3 different MTurk workers, who are required to have great than 80% `HIT Approval Rate` to improve the quality of answers. Figure 6 is an example of our evaluation template. We collect the answers and count the majority votes, where the majority vote is defined as model i has 2 or more votes (since we have 3 answers per question). We ignore the questions without majority votes. Finally, we collected 1,120 valid votes for 370 questions, in which 275 *relevance* questions, 281 *coherence* questions and 266 *informativeness* questions have majority votes.

[9] https://github.com/NLPCode/CBART
[10] https://www.mturk.com

REFERENCES

[1] Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*. 65–72.

[2] William Chan, Nikita Kitaev, Kelvin Guu, Mitchell Stern, and Jakob Uszkoreit. 2019. KERMIT: Generative Insertion-Based Modeling for Sequences. *ArXiv* abs/1906.01604 (2019).

[3] Li Chen and Feng Wang. 2017. Explaining recommendations based on feature sentiments in product reviews. (2017), 17–28.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. (2019), 4171–4186.

[5] Li Dong, Shaohan Huang, Furu Wei, Mirella Lapata, Ming Zhou, and Ke Xu. 2017. 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*. 623–632.

[6] Jingyue Gao, Xiting Wang, Yasha Wang, and Xing Xie. 2019. Explainable recommendation through attentive multi-view learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 3622–3629.

[7] Jiatao Gu, Qi Liu, and Kyunghyun Cho. 2019. Insertion-based Decoding with Automatically Inferred Generation Order. *Transactions of the Association for Computational Linguistics* 7 (2019), 661–676.

[8] Jiatao Gu, Changhan Wang, and Junbo Zhao. 2019. Levenshtein transformer. *Advances in Neural Information Processing Systems* 32.

[9] Deepesh V Hada and Shirish K Shevade. 2021. Rexplug: Explainable recommendation using plug-and-play language model. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 81–91.

[10] Xingwei He. 2021. Parallel Refinements for Lexically Constrained Text Generation with BART. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. 8653–8666.

[11] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian Error Linear Units (GELUs). *arXiv: Learning* (2016).

[12] Chris Hokamp and Qun Liu. 2017. Lexically Constrained Decoding for Sequence Generation Using Grid Beam Search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 1535–1546.

[13] J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 839–850.

[14] X Hua and L Wang. 2019. Sentence-Level Content Planning and Style Specification for Neural Text Generation. In *Conference on Empirical Methods in Natural Language Processing*.

[15] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 7871–7880.

[16] Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B Dolan. 2016. A Diversity-Promoting Objective Function for Neural Conversation Models. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 110–119.

[17] Junyi Li, Siqing Li, Wayne Xin Zhao, Gaole He, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. 2020. Knowledge-enhanced personalized review generation with capsule graph neural network. (2020), 735–744.

[18] Jiacheng Li, Jingbo Shang, and Julian McAuley. 2022. UCTopic: Unsupervised Contrastive Learning for Phrase Representations and Topic Mining. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Dublin, Ireland, 6159–6169. https://doi.org/10.18653/v1/2022.acl-long.426

[19] Junyi Li, Wayne Xin Zhao, Zhicheng Wei, Nicholas Jing Yuan, and Ji-Rong Wen. 2021. Knowledge-based review generation by coherence enhanced text planning. (2021), 183–192.

[20] Junyi Li, Wayne Xin Zhao, Ji-Rong Wen, and Yang Song. 2019. Generating Long and Informative Reviews with Aspect-Aware Coarse-to-Fine Decoding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. 1969–1979.

[21] Lei Li, Yongfeng Zhang, and Li Chen. 2020. Generate neural template explanations for recommendation. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*. 755–764.

[22] Lei Li, Yongfeng Zhang, and Li Chen. 2021. Personalized Transformer for Explainable Recommendation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*. 4947–4957.

[23] Lei Li, Yongfeng Zhang, and Li Chen. 2023. Personalized prompt learning for explainable recommendation. *ACM Transactions on Information Systems* 41, 4 (2023), 1–26.

[24] Piji Li, Zihao Wang, Zhaochun Ren, Lidong Bing, and Wai Lam. 2017. 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*. 345–354.

[25] Chin-Yew Lin. 2004. ROUGE: A Package for Automatic Evaluation of Summaries. In *Text summarization branches out*. 74–81.

[26] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv* abs/1907.11692 (2019).

[27] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*. https://openreview.net/forum?id=Bkg6RiCqY7

[28] Yichao Lu, Ruihai Dong, and Barry Smyth. 2018. Why I like it: multi-task learning for recommendation and explanation. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 4–12.

[29] Julian John McAuley and Jure Leskovec. 2013. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. (2013), 897–908.

[30] Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 6834–6842.

[31] Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-Step: Separating Planning from Realization in Neural Data-to-Text Generation. (2019), 2267–2277.

[32] Jianmo Ni, Jiacheng Li, and Julian McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)*. 188–197.

[33] Jianmo Ni, Zachary C Lipton, Sharad Vikram, and Julian McAuley. 2017. 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)*. 783–791.

[34] Jianmo Ni and Julian McAuley. 2018. Personalized review generation by expanding phrases and attending on aspect-aware representations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 706–711.

[35] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

[36] Matt Post and David Vilar. 2018. Fast Lexically Constrained Decoding with Dynamic Beam Allocation for Neural Machine Translation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*. 1314–1324.

[37] Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. In *International Conference on Machine Learning*. PMLR, 5976–5985.

[38] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. *Advances in neural information processing systems* 27.

[39] Jian Tang, Yifan Yang, Samuel Carton, Ming Zhang, and Qiaozhu Mei. 2016. Context-aware Natural Language Generation with Recurrent Neural Networks. *ArXiv* abs/1611.09900 (2016).

[40] Sean Welleck, Kianté Brantley, Hal Daumé Iii, and Kyunghyun Cho. 2019. Non-monotonic sequential text generation. In *International Conference on Machine Learning*. PMLR, 6716–6726.

[41] An Yan, Zhankui He, Jiacheng Li, Tianyang Zhang, and Julian McAuley. 2023. Personalized Showcases: Generating multi-modal explanations for recommendations. In *SIGIR*.

[42] Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. 2020. BERTScore: Evaluating Text Generation with BERT. *ICLR* (2020).

[43] Yongfeng Zhang, Xu Chen, et al. 2020. Explainable recommendation: A survey and new perspectives. *Foundations and Trends® in Information Retrieval* 14, 1 (2020), 1–101.

[44] Yongfeng Zhang, Guokun Lai, Min Zhang, Yi Zhang, Yiqun Liu, and Shaoping Ma. 2014. Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In *Proceedings of the 37th international ACM SIGIR conference on Research & development in information retrieval*. 83–92.

[45] Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and William B Dolan. 2020. POINTER: Constrained Progressive Text Generation via Insertion-based Generative Pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 8649–8670.