4-2022

AmpSum: adaptive multiple-product summarization towards improving recommendation captions

Quoc Tuan TRUONG
Singapore Management University, qttruong.2017@phdis.smu.edu.sg

Hady Wirawan LAUW
Singapore Management University, hadywlauw@smu.edu.sg

Changhe YUAN

Jin LI

Jim CHAN

See next page for additional authors

Citation

TRUONG, Quoc Tuan; LAUW, Hady Wirawan; YUAN, Changhe; LI, Jin; CHAN, Jim; PANTEL, Soo-Min; and LAUW, Hady W.. AmpSum: adaptive multiple-product summarization towards improving recommendation captions. (2022). *Proceedings of the 2022 ACM Web Conference, Lyon, France, April 25 - 29*. 2978-2988.
Available at: https://ink.library.smu.edu.sg/sis_research/7662

AmpSum: Adaptive Multiple-Product Summarization towards Improving Recommendation Captions

Quoc-Tuan Truong[1], Tong Zhao[2], Changhe Yuan[2], Jin Li[2], Jim Chan[2], Soo-Min Pantel[2], Hady W. Lauw[1]
{qttruong.2017,hadywlauw}@smu.edu.sg,{zhaoton,ychanghe,jincli,jamchan,pantel}@amazon.com
[1]Singapore Management University [2]Amazon, United States

ABSTRACT

In e-commerce websites, multiple related product recommendations are usually organized into "widgets", each given a name, as a recommendation caption, to describe the products within. These recommendation captions are usually manually crafted and generic in nature, making it difficult to attach meaningful and informative names at scale. As a result, the captions are inadequate in helping customers to better understand the connection between the multiple recommendations and make faster product discovery.

We propose an Adaptive Multiple-Product Summarization framework (AmpSum) that automatically and adaptively generates widget captions based on different recommended products. The multiplicity of products to be summarized in a widget caption is particularly novel. The lack of well-developed labels motivates us to design a weakly supervised learning approach with distant supervision to bootstrap the model learning from pseudo labels, and then fine-tune the model with a small amount of manual labels. To validate the efficacy of this method, we conduct extensive experiments on several product categories of Amazon data. The results demonstrate that our proposed framework consistently outperforms state-of-the-art baselines over 9.47-29.14% on ROUGE and 27.31% on METEOR. With case studies, we illustrate how AmpSum could adaptively generate summarization based on different product recommendations.

KEYWORDS

Product Summarization, Multiple-Product Summarization, Recommendation Captions

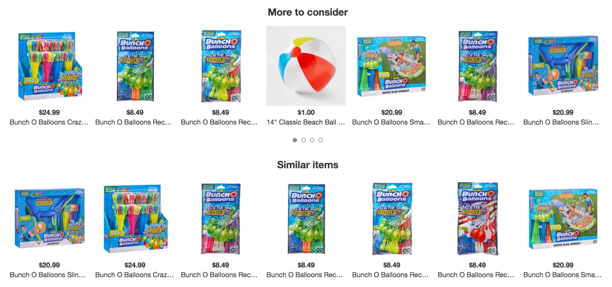
ACM Reference Format:
Quoc-Tuan Truong[1], Tong Zhao[2], Changhe Yuan[2], Jin Li[2],, Jim Chan[2], Soo-Min Pantel[2], Hady W. Lauw[1] . 2022. AmpSum: Adaptive Multiple-Product Summarization towards Improving Recommendation Captions . In *Proceedings of the ACM Web Conference 2022 (WWW '22), April 25–29, 2022, Virtual Event, Lyon, France.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3485447.3512018

1 INTRODUCTION

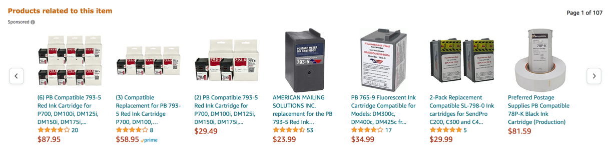
Recommender systems in e-commerce are important to help customers quickly discover their preferred products. The two prongs of facilitating this discovery are both the accuracy with which a recommender system could predict the likely preferred items, as well as the presentation of recommended items to the end consumers in an accessible and interpretable manner. Consequently, explainable recommendation [3, 8, 9, 14, 18, 39, 44, 45, 47] is fast gaining traction in academia and industry. However, most of the work [37, 41] target improving product-level recommendation accuracy (the first prong), and relatively few dive deep into the connections between multiple recommendations (the second prong), which is the focus of this work.

As far as explanations are concerned, the preponderance of academic literature is still on explaining a single product recommendation, anticipating that each product being recommended requires a separate explanation [44]. While that is a valid application scenario, an alternative scenario common in modern e-commerce platforms is where a recommender system organizes product recommendations into different "widgets", each consisting of products that are being recommended as a collective simultaneously because of their commonality (e.g., similar purpose) as well as their variability (e.g., different features). Examples of such widgets are shown in Figure 1.

Motivation. This work does not focus on identifying products to make up a widget recommendation per se. That presumably would have come from some underlying recommender system, to which we are agnostic. Rather, our focus is exclusively on generating a caption as explanation for a widget recommendation, in the form of a widget caption/name. Naming is crucial as it is a fleeting



(a) Recommendation widgets from Target.


Products related to this item
Sponsored
Page 1 of 107
(6) PB Compatible 793-5 Red Ink Cartridge for P700, DM100i, DM125i, DM150i, DM175i,…
20
$87.95
(3) Compatible Replacement for PB 793-5 Red Ink Cartridge P700, DM100,…
8
$58.95 prime
(2) PB Compatible 793-5 Red Ink Cartridge for P700, DM100i, DM125i, DM150i, DM175i,…
$29.49
AMERICAN MAILING SOLUTIONS INC. replacement for the PB 793-5 Red Ink…
53
$23.99
PB 765-9 Fluorescent Ink Cartridge Compatible for Models: DM300c, DM400c, DM425c fr…
17
$34.99
2-Pack Replacement Compatible SL-798-0 Ink cartridges for SendPro C200, C300 and C4…
5
$29.99
Preferred Postage Supplies PB Compatible 78P-K Black Ink Cartridge (Production)
$81.59

(b) Recommendation widgets from Amazon.

Figure 1: Examples of recommendation widgets with general captions on various e-commerce websites.

WWW '22, April 25–29, 2022, Virtual Event, Lyon, France
© 2022 Association for Computing Machinery.
ACM ISBN 978-1-4503-9096-5/22/04…$15.00
https://doi.org/10.1145/3485447.3512018

opportunity to attract user's attention by providing what is effectively a mere glance of the products within. A good name entices a user to look into individual products. A lesser name may result in a whole collection of recommended products being ignored or bypassed altogether.

As evident from Figure 1, we observe three major issues with widget captions currently found online. First, widget captions such as *"More to consider"* and *"Product related to this item"* are generic and uninformative about what constitutes important product information. Second, most widget captions are presumably manually crafted since there is so little variation. There is no dynamic adaptation with the change in content, i.e., the same widget caption might be used even for a completely different set of products. Third, we observe useful information from product titles, but such information may not surface to the user. While technically a widget may show a subset of products, long titles are often truncated rendering important distinguishing information invisible to the user.

Problem. We postulate that due to the multiplicity of products within a widget, we need a widget caption/name as a compact form of representation of the collective. It is thus natural to formulate the task of widget caption generation as a multiple-product summarization problem. We expect to solve the problem in a scalable and adaptive way, so as to automatically generate relevant summaries based on a particular set of products given within a widget. For example, given a set of recommended TV products with different brands and display sizes, a good widget caption may be *"Explore more TVs with different brands and display sizes"*. If the recommended TVs are all of the same display size, e.g., 55", but from different brands, a good widget caption could be *"Consider more 55" TVs from different brands"*. Such adaptive customisability requires the method to have the capability to identify important attributes from recommended products, as well as to decide how to highlight the common and distinguishing properties among the products.

At a glance, this problem is reminiscent of multiple-document summarization [2, 11, 22, 46]. However, existing multiple-document summarization task creates a short summary from a set of unstructured textual documents by ranking textual segments (usually sentences) according to the relationship (e.g., similarity, diversity) to other segments and to other documents [37, 46]. In our case, there is a need to leverage product attributes as guidance to conduct summarization in e-commerce domain. We will elaborate on this distinction further in the related work (Section 5), as well as include a state-of-the-art summarization technique as a baseline.

Approach and Contributions. Solving multiple-product summarization problem is non-trivial and requires specific consideration due to the nature of products. Firstly, different product categories have different important properties/attributes, e.g., display size for TV vs. roast type for coffee. Secondly, even within a product category, different recommended contents may need distinct summaries to highlight the important common and different properties on the recommendations. For example, multiple TV recommendations from "Samsung" should highlight their identical brand information in the summary, while TV recommendations from different brands but with similar display sizes should highlight both similar display size and different brand information in the summary. Last but not least, as the multiple-product summarization is novel for both academia and industry, we pay attention to minimizing the human labeling efforts to collect training samples to solve the problem not only at scale but also adaptively.

One key insight that motivates our general approach is how beneficial it would be to first summarize each product on its own, before summarizing these single-product summaries further into a multiple-product summary. For one benefit, the single-product summarization based on title, keywords and attributes helps in extracting important properties. For another benefit, the more concise product titles can be shown even as part of the limited space given to a widget, enhancing the quality of information surfaced to the user. In turn, the multiple-product summarization could adaptively build up a merged summarization, which highlights the common and different attributes among the specific products within a widget.

In summary, our contributions are three-fold:

- *Problem Formulation.* First, to the best of our awareness, the problem of multiple-product summarization is not widely-studied in the literature. Thus, our problem formulation with the novel approach of two tasks is one of our contributions.
- *Methodology.* Second, we design a new framework AmpSum consisting of two components tailored for the formulated tasks. In addition, we propose to overcome the challenge of limited human labels with weakly supervised learning where we introduce ways to derive pseudo labels from product catalog features.
- *Performance.* Third, we systematically evaluate the efficacy of AmpSum on multiple datasets from Amazon.com. Not only do we evaluate this quantitatively, but we also provide case studies that illustrate practicality of the proposed framework.

2 PRELIMINARIES

In this section, we begin with our problem setup in which we formulate two tasks to be solved. We then go into details in describing how to obtain distant supervision labels to bootstrap model learning in the face of limited human labels.

Problem Formulation. The problem of multiple-product summarization problem towards improving recommendation explainability is formulated as follows. Given a sequence of recommended products $\mathcal{P} = [\mathbf{P}_1, \ldots, \mathbf{P}_K]$, each product $\mathbf{P}_i$ consists of *catalog features* C_i (including title, product type, product attributes) that forms product text $\mathbf{X}_i$, we would like to learn a model $\mathcal{M}$ which is capable of producing summary $\mathbf{Y}$ serving as title string for the recommendation widget. We propose to solve the problem by breaking it down into two subsequent tasks that can be stated as the following.

Problem 2.1 (Single-Product Summarization). *Given a product* $\mathbf{P}$ *with its catalog features* C *forming a text document* $\mathbf{X}$*, we seek to learn a model* $\mathcal{M}_1$ *which can produce a summary* $\mathbf{Y}$ *for* $\mathbf{P}$*. The summary* $\mathbf{Y}$ *carries essential information of* $\mathbf{P}$ *serving as an input to Problem 2.2. Also,* $\mathbf{Y}$ *is concise enough to be the title of* $\mathbf{P}$ *in a recommendation widget.*

Problem 2.2 (Multiple-Product Summarization). *Given a sequence of recommended products* $\mathcal{P} = [\mathbf{P}_1, \ldots, \mathbf{P}_K]$ *with their corresponding summaries* $[\mathbf{Y}_1, \ldots, \mathbf{Y}_K]$ *output by* $\mathcal{M}_1$*, we seek to learn a model* $\mathcal{M}_2$ *which can produce a summary* $\mathbf{Y}_{K+1}$ *being the caption for the recommendation widget. The summary* $\mathbf{Y}_{K+1}$ *serves as a form of explanation to users received the recommendations.*

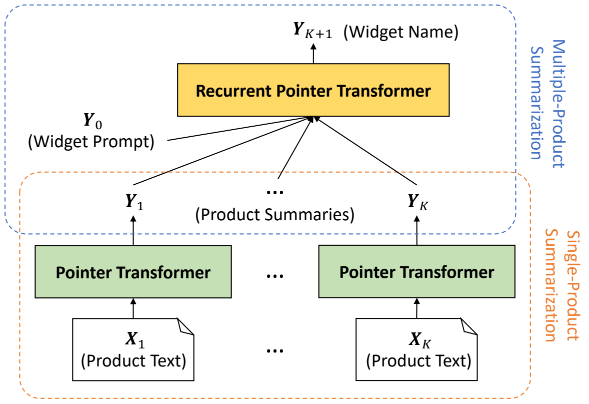


Figure 2: Overall architecture of AmpSum.

Distant Supervision Labels. As discussed in Section 1, one critical challenge to multiple-product summarization problem is limited human labels, as such a summarization problem in e-commerce is quite new and no established datasets can be directly used. We address the problem via weakly supervised learning with distant supervision. The key idea is to arrive at heuristic rules that can generate imperfect yet plentiful pseudo labels. Such pseudo labels will be used to bootstrap the model learning, while then a small number of human labels will be used to further fine-tune the model.

First, pseudo labels for single-product summarization follow the following format: *<Initial Part> <Property Part>*. The *<Initial Part>*, which contains basic descriptive keywords of a product (e.g., brand name, model variant, main features), can be constructed via uni-gram matching of the original title with catalog attributes and product taxonomy. The *<Property Part>*, which is essential to the second task, covers important attributes (e.g., material and size for fashions, display size for TVs) of the product that need to be highlighted. The important product attributes can be mined from user search queries.

Second, pseudo labels for multiple-product summarization follow the template: *<Similarity> <Difference>*. Based on the summaries generated from single-product summarization, *<Similarity>* highlights the common parts among products while *<Difference>* covering the disjoint/distinguishable attributes. Both can be extracted with some simple heuristics to form the pseudo labels. These labels are used for model learning while the output summaries can be further post-processed (i.e., added template words) to produce the final widget captions.

With the discussed approaches for generating pseudo labels, we later construct multiple datasets for our experiments (Section 4), in addition to a set of human labels for each task.

3 AMPSUM: ADAPTIVE MULTIPLE-PRODUCT SUMMARIZATION

We now describe our framework *AmpSum*, consisting of two main components: (1) *Pointer Transformer* module for performing single-product summarization and (2) *Recurrent Pointer Transformer* module for performing multiple-product summarization. Figure 2 illustrates the overall architecture. For the ease of reference, the main notations used in this paper are described in Table 1.

Table 1: Notations.

Symbol	Description
C_i	Product catalog features
$\mathbf{X}_i$	Product text document
$\mathbf{Y}_0$	Widget prompt
$\mathbf{Y}_{1,\ldots,K}$	Single-product summaries
$\mathbf{Y}_{K+1}$	Multiple-product summary/widget caption
$\mathbf{W}_*, \mathbf{b}_*$	Learnable weight matrices and bias vectors
$\mathbf{U}_i, \mathbf{u}_t$	Intermediate representations from `transformer_encoder`
$\mathbf{H}_i, \mathbf{h}_t$	Intermediate representations from `transformer_decoder`
$\mathbf{R}_i$	Intermediate representations from GRU
$\mathbf{C}, \mathbf{c}_t$	Copy distributions from pointer networks

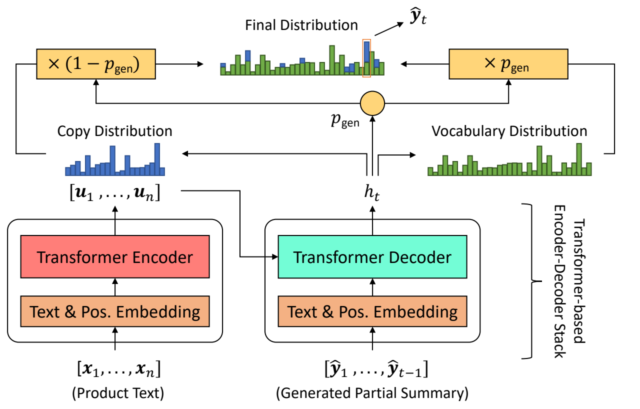


Figure 3: Pointer Transformer module. The architecture consists of two layers, a Transformer-based Encoder-Decoder Stack and a Pointer Network, enabling a soft copy mechanism during the decoding process.

3.1 Single-Product Summarization with Pointer Transformer

The objective of single-product summarization is to take a product text $\mathbf{X} = [\mathbf{x}_1; \ldots; \mathbf{x}_n]^T$ as input, and output a summary $\mathbf{Y} = [\mathbf{y}_1; \ldots; \mathbf{y}_m]^T$ for the product of interest. The input $\mathbf{X}$ is constructed from product catalog features C (i.e., product title, product type, product attributes), where output $\mathbf{Y}$ is the expected concise summary highlighting important properties of the product.

To tackle this summarization problem, we design a module named Pointer Transformer (Figure 3). At the base layer, it employs the standard sequence-to-sequence Transformer-based architecture [33]. In particular, this base layer, which we refer to as Transformer-based Encoder-Decoder Stack, performs the following transformations:

$$\mathbf{U} = \texttt{transformer\_encoder}(\mathbf{X}\mathbf{W}_e + \mathbf{W}_p) \tag{1}$$

$$\mathbf{h}_t = \texttt{transformer\_decoder}(\mathbf{Y}_{1:t-1}\mathbf{W}_e + \mathbf{W}_p; \mathbf{U}) \tag{2}$$

where $\mathbf{W}_e \in \mathbb{R}^{V \times d}$ and $\mathbf{W}_p \in \mathbb{R}^{n \times d}$ are the token embedding matrix and positional embedding matrix with V being the size of vocabulary, n is the length of input text, and d is the hidden dimensions of our model, $\mathbf{U} = [\mathbf{u}_1; \ldots; \mathbf{u}_n]^T, \mathbf{U} \in \mathbb{R}^{n \times d}$ is the output of the encoder block used for cross-attention in the decoder, and $\mathbf{h}_t \in \mathbb{R}^d$ is the hidden representation output by the decoder block at timestep t.

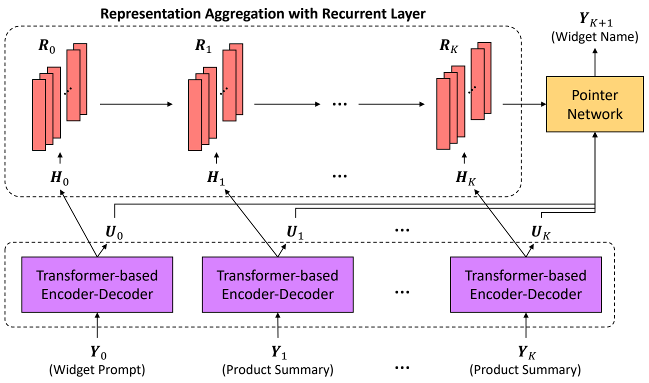

Representation Aggregation with Recurrent Layer
R_0
R_1
R_K
H_0
H_1
H_K
Y_{K+1}
(Widget Name)
Pointer Network
U_0
U_1
U_K
Transformer-based Encoder-Decoder
Transformer-based Encoder-Decoder
Transformer-based Encoder-Decoder
Y_0
(Widget Prompt)
Y_1
(Product Summary)
Y_K
(Product Summary)

Figure 4: Recurrent Pointer Transformer module. The architecture consists of two layers, a Transformer-based Encoder-Decoder Stack and a Recurrent Layer for representation aggregation followed by a Pointer Network for decoding.

In the next layer of the Pointer Transformer, we design a Pointer Network inspired by various successes in sequence-to-sequence modeling [34] as well as summarization [31], allowing the model to both copy tokens via pointing to the input and generate tokens from a pre-defined vocabulary. Given hidden representation $\mathbf{h}_t$ from the previous layer,

$$p_{\text{gen}} = \sigma(\mathbf{w}_{\text{gen}}^T \mathbf{h}_t + b_{\text{gen}}) \tag{3}$$

is the probability for the next token $\mathbf{y}_t$ being generated from the vocabulary, where $p_{\text{gen}} \in [0, 1]$ with σ being the $\texttt{sigmoid}$ function and $\mathbf{w}_{\text{gen}} \in \mathbb{R}^d, b_{\text{gen}}$ are learnable weight vector and bias scalar. We obtain the final distribution over tokens as follows:

$$P(y) = p_{\text{gen}} P_{\text{vocab}}(y) + (1 - p_{\text{gen}}) \sum_{i:x_i=y} c_t^i \tag{4}$$

where $P_{\text{vocab}} = \texttt{softmax}(\mathbf{W}_v \mathbf{h}_t + \mathbf{b}_v)$ and c_t^i is the probability of token i in the input sequence being copied at timestep t. The probability distribution $\mathbf{c}_t = [c_t^1, \ldots, c_t^n]^T$, which we refer to as the copy distribution (Figure 3), is calculated via normalized scaled dot-products:

$$\mathbf{c}_t = \texttt{softmax}(\mathbf{K}\mathbf{q}_t / \sqrt{d}) \ ; \quad \mathbf{q}_t = \mathbf{h}_t^T \mathbf{W}_q \ ; \quad \mathbf{K} = \mathbf{U}\mathbf{W}_k \tag{5}$$

where $\mathbf{W}_q \in \mathbb{R}^{d \times d}, \mathbf{W}_k \in \mathbb{R}^{d \times d}, \mathbf{W}_v \in \mathbb{R}^{V \times d}, \mathbf{b}_v \in \mathbb{R}^V$ are learnable projection matrices and bias vector.

Learning parameters of the model is done via minimizing the negative log likelihood of the most probable token y_t^* at each timestep t, and the loss for the whole m-length summary is:

$$\text{loss} = -\frac{1}{m} \sum_{t=1}^{m} \log P(y_t^*) \tag{6}$$

3.2 Multiple-Product Summarization with Recurrent Pointer Transformer

We now describe our modeling approach for the second task. Given a set of recommended products with their corresponding summaries, our goal is to arrive at a concise yet informative multiple-product summary serving as the recommendation widget caption. Based on the nature of the recommendations that come in a sequential order indicating their 'recommendation-relevant' scores, we design a module named Recurrent Pointer Transformer (Figure 4) that can generate multiple-product summary while taking into account such sequential information. In addition, to make our method more flexible and adaptive to different forms of recommendations, we introduce a concept of *widget prompt* serving as a form of contextual information being input to the Recurrent Pointer Transformer module. Widget prompt could be a set of product attributes that we would like the model to pay attention to, or a template of the widget caption to be followed. In our experiments, widget prompt is set to be product type, which is suitable for substitute recommendations, combined with important attributes of such product type that are mined from user search queries.

Suppose that there are K recommended products in a widget, input to the model will include widget prompt $\mathbf{Y}_0$ and product summaries $[\mathbf{Y}_1; \ldots; \mathbf{Y}_K]$ from the previous task. A widget caption to be generated $\mathbf{Y}_{K+1}$ has length of m tokens. We obtain intermediate hidden representation matrices via the first layer:

$$[\mathbf{U}_i, \mathbf{H}_i] = \texttt{transformer}(\mathbf{Y}_i), \ i \in [0, \ldots, K] \tag{7}$$

where $\mathbf{U}_i \in \mathbb{R}^{n \times d}$ and $\mathbf{H}_i \in \mathbb{R}^{m \times d}$ are outputs of encoder and decoder, respectively.

To retain sequential order of the recommendations, a recurrent layer with Gated Recurrent Unit (GRU) [5] is employed for representation aggregation, before the decoding step. In particular, the recurrent layer takes $\mathbf{H}_i$ as input and output $\mathbf{R}_i$ at each recurrent step (i.e., each product in the sequence):

$$\mathbf{R}_i = \texttt{GRU}(\mathbf{H}_i), \ i \in [0, \ldots, K] \tag{8}$$

The last output $\mathbf{R}_K \in \mathbb{R}^{m \times d}$ of recurrent layer serves as final representation of the widget. It then becomes the input to the Pointer Network for decoding. While the decoding process with soft copy mechanism is similar to Pointer Transformer detailed earlier, instead of deriving formulas for each timestep (i.e., each token), here we present the computation in matrix notations:

$$[p_{\text{gen}}^1, \ldots, p_{\text{gen}}^m]^T = \sigma(\mathbf{R}_K \mathbf{w}_{\text{gen}} + b_{\text{gen}}) \tag{9}$$

$$[P_{\text{vocab}}^1; \ldots; P_{\text{vocab}}^m] = \texttt{softmax}(\mathbf{W}_v \mathbf{R}_K^T + \mathbf{b}_v) \tag{10}$$

The matrix $\mathbf{C} = [\mathbf{c}_1; \ldots; \mathbf{c}_m]^T, \mathbf{C} \in \mathbb{R}^{m \times n(K+1)}$ of copy distributions, is calculated as follows:

$$\mathbf{C} = \texttt{softmax}(\mathbf{Q}\mathbf{K}^T / \sqrt{d}) \ ; \quad \mathbf{Q} = \mathbf{R}_K \mathbf{W}_q \ ; \quad \mathbf{K} = \begin{bmatrix} \mathbf{U}_0 \\ \vdots \\ \mathbf{U}_K \end{bmatrix} \mathbf{W}_k \tag{11}$$

With all the quantities in place, the final distributions over tokens can be obtained similarly to Eqn. 4, and parameters optimization is via minimizing the loss function in Eqn. 6 for each widget caption.

3.3 Model Details

Learning. For the implementation of Transformer-based Encoder-Decoder Stack, we follows the base architecture (6 layers for both encoder and decoder, hidden size of 768, max input length of 1024) of BART [19], with a difference is in the change of activation functions from $\texttt{ReLU}$ to $\texttt{GeLU}$. During training, parts of the weights of the Pointer Transformer module are initialized from the pre-trained BART Conditional Generation model, while the weights of pointer network are initialized randomly. For the Recurrent Pointer Transformer, weights are initialized from the trained Pointer Transformer,

Algorithm 1 AmpSum Inference Procedure

```
Input: recommended products $\mathcal{P} = [\mathbf{P}_1, \ldots, \mathbf{P}_K]$
Output: product summaries S, widget caption WC
1: initialization
2:     S = [], WC = None
3: for $\mathrm{P}_i \in \mathcal{P}$ do
4:     Obtain product catalog features $C_i$
5:     Prepare product text document $\mathbf{X}_i$
6:     $\mathrm{Y}_i$ = pointer_transformer($\mathbf{X}_i$)
7:     S.append($\mathrm{Y}_i$)
8: end for
9: Prepare widget prompt $\mathrm{Y}_0$
10: WC = recurrent_pointer_transformer($\mathrm{Y}_0$, S)
11: return {S, WC}
```

except for the recurrent layer with GRU initialized randomly. We perform early stopping using validation set and observe that our model quickly converges after 3-5 epochs for both tasks, with batch size of 16 and other optimization hyper-parameters held to the recommended values by Lewis et al. [19].

Inference. Given a sequence of recommended products $\mathcal{P}$, to generate a widget caption, the first step is to obtain summaries for all of the products using the Pointer Transformer module. The summaries are then paired with a widget prompt serving as input for the second step of widget caption generation using the Recurrent Pointer Transfomer module. Note that both single-product summaries and widget caption are the final outputs of our framework. The maximum length of decoding text is set to $m = 64$ as we would not want our summaries to be very long, where beam search [1] is performed with a beam width of 4. The complete inference procedure is described in Algorithm 1.

4 EXPERIMENTS

Our experimental objective is to investigate the effectiveness of the proposed framework *AmpSum* in producing summaries serving as neat title strings for recommended products and widgets. We systematically evaluate the model performance on two tasks, single-product summarization and multiple-product summarization.

4.1 Experimental Setups

Datasets. Our experiments are conducted on three real-world datasets of different categories collected from *Amazon.com* including Fashion, Electronics, and Office. In addition to the pseudo labels derived from distant supervision approach, we obtain a number of human labels for each task for further fine-tuning and evaluation of the models regarding human quality. Statistics of the datasets are reported in Table 2. For each dataset, we randomly split it into training set (60%), validation set (20%), and test set (20%). Best settings of model hyper-parameters (i.e., number of epochs and batch size) are searched on the validation set, while during the fine-tuning, the best settings are reused. Thus, we split the human labels with the ratio of 60% and 40% for fine-tuning and evaluation, respectively.

Metrics. To evaluate the performance of comparative methods, we employ two commonly used summarization metrics:

[1]Beam search is implemented using a greedy strategy over a fixed K number of beams (top-K most probable decoded sequences).

Table 2: Dataset statistics.

	Fashion	Electronics	Office	Human
# Products	115K	78K	57K	898
# Widgets	131K	91K	67K	396

- **ROUGE** [21] is the most standard metric for measuring quality of summaries for the summarization problem. We report the F1-scores of: ROUGE-1 (R-1) measuring the unigram-overlap, ROUGE-2 (R-2) measuring the bigram-overlap, and ROUGE-L (R-L) measuring the longest sequence between the reference summary and the evaluated summary.
- **METEOR** [1] is developed for evaluating the quality of generated text in the translation problem, though it is also widely used for the problem of summarization. We report METEOR score of the full mode which rewards matching stems, synonyms, and paraphrases.

Baselines. There are two tasks in our problem. For the first task, single-product summarization, we compare our model with the following baselines:

- **Original Title** is the baseline using the product titles currently being displayed on the Amazon website. We use it as a ground to all of the learning methods trying to improve the summarization quality.
- **Seq2Seq+Attn** [26] employs sequence-to-sequence encoder-decoder framework to tackle abstractive summarization. The architecture uses bi-LSTM as the encoder for learning abstract representation of the input text, and the decoder consists of LSTM with attention mechanism generating word probabilities through time.
- **PTGEN** [31] is a competitive method for text summarization. The proposed architecture improves upon Seq2Seq+Attn by introducing a pointer generator, on top of the attention distribution, being capable of copying words from the source text via a soft copy mechanism.
- **PTGEN+Cov** [31] is an improved version of PTGEN with the integration of coverage loss penalizing repeatedly attending to the same locations in the original model.
- **BART** [19] is the state-of-the-art method for text summarization. It is based on the Transformer sequence-to-sequence model where the encoder is similar to the BERT model and the decoder and training objective are similar to the GPT model. We use the base architecture of BART (i.e., 6 layers for both encoder and decoder, hidden size of 768).

For the second task, multiple-product summarization, we adapt the state-of-the-art method BART into a hierarchical structure, thus we name it Hierarchical BART (HBART):

- **HBART** concatenates outputs of BART [19] from single-product summarization task, it then uses that concatenated text as input to another BART module to generate a multiple-product summary.
- **HBART+Perm** permutes the order of products in the input sequences for generating more sequential variants. It can be seen as a data augmentation technique, which helps learn more robust representation of the set of products.

Table 3: Single-product summarization on pseudo labels (higher is better).

	Fashion				Electronics				Office			
	R-1	R-2	R-L	METEOR	R-1	R-2	R-L	METEOR	R-1	R-2	R-L	METEOR
Original Title	54.41	26.46	49.42	43.91	45.67	22.92	40.51	43.94	42.51	15.20	38.34	44.40
Seq2Seq+Attn	94.02	90.73	93.90	95.91	86.92	79.66	86.73	89.11	81.74	78.89	81.56	84.15
PTGEN	94.80	91.83	94.66	96.87	90.84	88.94	90.74	94.94	87.10	84.36	86.99	89.43
PTGEN+Cov	95.26	91.33	95.15	97.11	91.90	90.10	91.87	96.13	89.21	86.42	89.14	91.63
BART	99.25	98.36	99.20	99.36	99.18	98.47	99.17	98.93	98.35	92.76	96.64	96.91
AmpSum	**99.47**	**98.87**	**99.43**	**99.46**	**99.39**	**98.86**	**99.38**	**99.07**	**99.09**	**94.12**	**97.56**	**97.10**

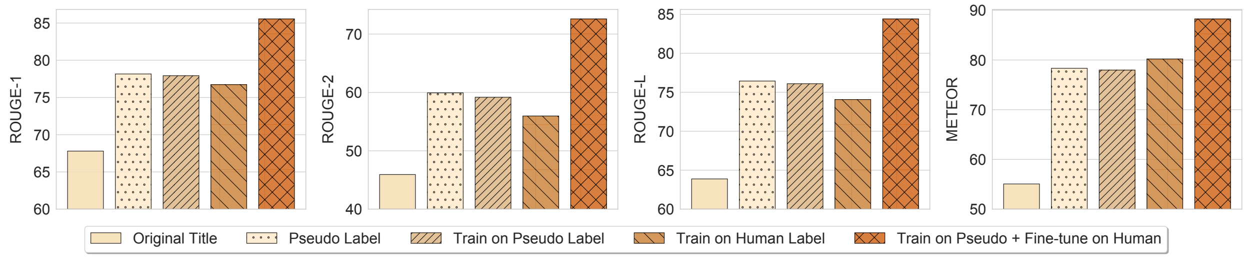


Figure 5: Ablation analysis: weakly supervised learning with distant supervision (evaluated on human labels).

Table 4: Single-product summarization on human labels.

	R-1	R-2	R-L	METEOR
Original Title	67.80	45.93	63.88	55.07
Pseudo Label	78.16	59.93	76.44	78.33
Seq2Seq+Attn	69.33	51.60	68.69	74.03
PTGEN	79.94	64.62	78.97	83.55
PTGEN+Cov	80.44	64.84	79.13	84.43
BART	84.67	71.01	83.38	87.78
AmpSum	**85.55**	**72.60**	**84.41**	**88.26**
Improvement	1.04%	2.24%	1.24%	0.54%

4.2 Task #1: Single-Product Summarization

In Table 3, we report the experimental results of all compared methods on the pseudo labels. Generally, all the learning methods show decent performance. This is expected as all the models having access to the information being used to generate the pseudo labels. Therefore, the models should be able to learn to extract useful terms from the input in order to form the correct summarization output. Notably, *BART* and *AmpSum*, both are Transformer-based models, demonstrate superior performances as compared to other baselines, especially on Fashion and Electronics datasets. Only *Original Title* is far behind as the current titles are generally longer and off the format of concise single-product summaries.

Table 4 shows the results of all models after being fine-tuned on human labels. Again, *Original Title* and *Pseudo Label* are included as reference grounds. For the learning models, after being fine-tuned, most of them perform better than *Pseudo Label*, especially *AmpSum*. One can see that *AmpSum*'s performance is better than all compared methods, and the improvement of *AmpSum* over the second best baseline *BART* is better observed than the evaluation on pseudo labels. Noted that achieving good performance on human labels is more difficult as compared to the pseudo labels setting. These results, both on pseudo labels and human labels, evidently demonstrate that with the augmentation of pointer module on top of transformer networks, the proposed *Pointer Transformer* module is more effective to generate accurate single-product summaries.

Original Title: Under Armour Girls' Grade School KickIt2 Splatter Mid Sneaker, Black (001) /White, 7
AmpSum: *Under Armour Girls' Sneaker (Synthetic Rubber, Black/White, 7)*

Original Title: COCOSHIP Turquoise & Orange Tangerine Fruit Retro Tie Front One Piece Bather Swimsuit High Waist Cut Out Swimwear Maillot 14 (FBA)
AmpSum: *COCOSHIP Women's Tie Front One Piece Bather Swimsuit Retro High Waist Cut Out (Nylon, Tangerine, 14)*

Original Title: Sexy A Line Chiffon Evening Dresses 2019 Formal Ball Gown Manual Beaded Long Sleeves Scoop Neck Full Length Party Gown for Women Empire Waist Mother Dresses YRSH047 Navy Blue Size 12
AmpSum: *Women's Long Dress Beaded Long Sleeve (Chiffon, Navy Blue, 12)*

Original Title: YDN Women Tassels Formal Slip On Pumps Office Ladies Close Round Toe Block Mid Heeled Fashion Dress Shoes Fuchsia 8
AmpSum: *YDN Women Formal Slip On Shoes (Faux Leather, Fuchsia, 8)*

Original Title: Lito Angels Girls' Sequins Wedding Flower Girl Formal Dress Pageant Party Occasion Size 8 Pink
AmpSum: *Lito Angels Girls' Sequins Flower Formal Dress (Satin, Pink, 8)*

Figure 6: Examples of single-product summaries produced by *AmpSum* as compared to *Original Title*.

Ablation Analysis. One natural question to ask is that what the performance of *AmpSum* would be if the model is directly trained on human labels. In Figure 5, we report results of an experiment comparing different approaches for training *AmpSum* for single-product summarization. As we can observe, direct training on human labels results in a worse performance even compared to training on pseudo labels without further fine-tuning, in terms

Table 5: Multiple-product summarization on pseudo labels (higher is better).

	Fashion				Electronics				Office			
	R-1	R-2	R-L	METEOR	R-1	R-2	R-L	METEOR	R-1	R-2	R-L	METEOR
HBART	93.37	87.68	92.39	92.54	91.36	85.39	90.21	91.16	89.17	81.72	87.69	88.58
HBART+Perm	95.70	89.76	94.70	94.82	93.49	87.28	92.31	93.25	91.13	83.42	89.62	90.49
AmpSum (MaxPool)	97.22	96.93	97.21	96.92	96.65	96.35	96.65	96.53	94.11	91.46	94.09	94.28
AmpSum (AvgPool)	97.68	97.34	97.68	97.38	97.62	97.27	97.62	97.46	95.65	95.33	95.65	95.52
AmpSum (LSTM)	98.89	98.43	98.89	98.60	98.43	97.86	98.43	98.28	98.17	97.50	98.17	97.97
AmpSum (GRU)	**99.18**	**98.83**	**99.18**	**98.88**	**99.07**	**98.66**	**99.07**	**98.91**	**98.83**	**98.37**	**98.83**	**98.71**

Table 6: Multiple-product summarization on human labels.

	R-1	R-2	R-L	METEOR
HBART	66.69	39.70	61.36	50.22
HBART+Perm	68.33	39.85	62.44	51.44
AmpSum (MaxPool)	70.81	45.48	65.93	60.72
AmpSum (AvgPool)	71.64	45.97	66.63	61.34
AmpSum (LSTM)	73.11	46.53	68.79	63.49
AmpSum (GRU)	**74.79**	**51.47**	**71.30**	**65.49**
Improvement	9.47%	29.14%	14.19%	27.31%

of ROUGE scores. With limited number of human labels, it is very challenging to learn a good summarization model for the task. By bootstrapping the model training with the pseudo labels and then fine-tuning on human labels, *AmpSum* can perform much better as compared to solely training either on human labels or pseudo labels. This analysis emphasizes the importance and effectiveness of weakly supervised learning with distant supervision. The approach clearly helps addressing the challenge of limited human labels for product summarization task.

Case Studies. To provide better insights of *AmpSum* model via qualitative analysis, Figure 6 showcases several examples of generated single-product summaries, which are identical to human labels, compared to the original product titles. Generally, we find that the summaries by *AmpSum* are shorter, more concise, and more well-formed than the original titles. Therefore, they can be potential replacements of the current product titles displayed in the recommendation widgets mitigating the limited space issue. In addition, important attributes of products are highlighted and organized towards the end of the generated summaries making them suitable inputs for the second task.

4.3 Task #2: Multiple-Product Summarization

We now look at the performance of models on generating multiple-product summaries. Table 5 and Table 6 report the results evaluated on different sets of pseudo labels as well as human labels. Employing *BART* as the core building block, *HBART* establishes state-of-the-art baseline for our task at hand. By introducing a simple yet effective data permutation augmentation, *HBART+Perm* consistently improves upon *BART* in all comparisons. One possible explanation for such improvements is that *HBART+Perm* could learn a more robust positional encoding scheme than *HBART* thanks to the permutations. For the proposed *AmpSum* method, we compare

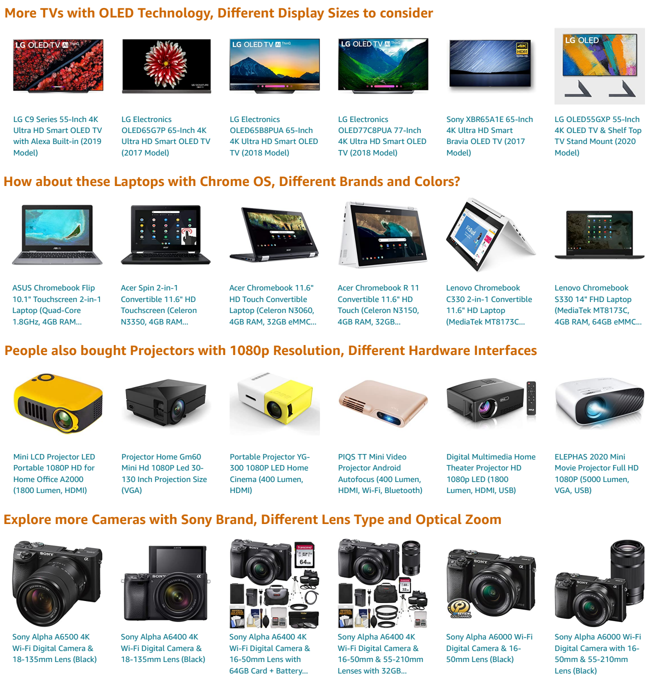

More TVs with OLED Technology, Different Display Sizes to consider
LG C9 Series 55-Inch 4K Ultra HD Smart OLED TV with Alexa Built-in (2019 Model)
LG Electronics OLED65G7P 65-Inch 4K Ultra HD Smart OLED TV (2017 Model)
LG Electronics OLED65B8PUA 65-Inch 4K Ultra HD Smart OLED TV (2018 Model)
LG Electronics OLED77C8PUA 77-Inch 4K Ultra HD Smart OLED TV (2018 Model)
Sony XBR65A1E 65-Inch 4K Ultra HD Smart Bravia OLED TV (2017 Model)
LG OLED55GXP 55-Inch 4K OLED TV & Shelf Top TV Stand Mount (2020 Model)
How about these Laptops with Chrome OS, Different Brands and Colors?
ASUS Chromebook Flip 10.1" Touchscreen 2-in-1 Laptop (Quad-Core 1.8GHz, 4GB RAM...
Acer Spin 2-in-1 Convertible 11.6" HD Touchscreen (Celeron N3350, 4GB RAM...
Acer Chromebook 11.6" HD Touch Convertible Laptop (Celeron N3060, 4GB RAM, 32GB eMMC...
Acer Chromebook R 11 Convertible 11.6" HD Touch (Celeron N3150, 4GB RAM, 32GB...
Lenovo Chromebook C330 2-in-1 Convertible 11.6" HD Laptop (MediaTek MT8173C...
Lenovo Chromebook S330 14" FHD Laptop (MediaTek MT8173C, 4GB RAM, 64GB eMMC...
People also bought Projectors with 1080p Resolution, Different Hardware Interfaces
Mini LCD Projector LED Portable 1080P HD for Home Office A2000 (1800 Lumen, HDMI)
Projector Home Gm60 Mini Hd 1080P Led 30-130 Inch Projection Size (VGA)
Portable Projector YG-300 1080P LED Home Cinema (400 Lumen, HDMI)
PIQS TT Mini Video Projector Android Autofocus (400 Lumen, HDMI, Wi-Fi, Bluetooth)
Digital Multimedia Home Theater Projector HD 1080p LED (1800 Lumen, HDMI, USB)
ELEPHAS 2020 Mini Movie Projector Full HD 1080P (5000 Lumen, VGA, USB)
Explore more Cameras with Sony Brand, Different Lens Type and Optical Zoom
Sony Alpha A6500 4K Wi-Fi Digital Camera & 18-135mm Lens (Black)
Sony Alpha A6400 4K Wi-Fi Digital Camera & 18-135mm Lens (Black)
Sony Alpha A6400 4K Wi-Fi Digital Camera & 16-50mm Lens with 64GB Card + Battery...
Sony Alpha A6400 4K Wi-Fi Digital Camera & 16-50mm & 55-210mm Lenses with 32GB...
Sony Alpha A6000 Wi-Fi Digital Camera & 16-50mm Lens (Black)
Sony Alpha A6000 Wi-Fi Digital Camera with 16-50mm & 55-210mm Lens (Black)

Figure 7: Examples of multiple-product summaries/widget captions (colored orange) produced by AmpSum. These widgets contain substitute recommendations and usually named *"Products related to this item"* or *"You may also like"*.

several model variants regarding different choices for the the sequence aggregation layer. The best performing variant is *AmpSum (GRU)* which achieves significant improvements over *HBART+Perm*, especially on human label evaluation denoted in Table 6 with 9.47%-29.14% on ROUGE scores and 27.31% on METEOR. These results vividly highlight the efficacy of the proposed *Recurrent Pointer Transformer* module for generating multiple-product summaries.

Ablation Analysis. Between *AmpSum (GRU)* and *AmpSum (LSTM)*, they are very comparable as both taking into account the sequential order of the recommendations in the widgets in the modeling approach. The advantage is clearly demonstrated via better performances as compared to *AmpSum (MaxPool)* and *AmpSum (AvgPool)* with simple aggregation functions. Nevertheless, *AmpSum (MaxPool)* and *AmpSum (AvgPool)* both perform better than *HBART* which simply concatenates the product summaries into a single document. We argue that *HBART* can not learn an effective encoding layer for multiple products via simple concatenation approach, in which *HBART+Perm* tries to improve with the permutation augmentation. This observation confirms the design choice of *Recurrent Pointer Transformer* module being suitable for our task of multiple-product summarization.

Case Studies. For a complete view on how the outputs by *AmpSum* would be used to improve the explainability as well as readability of recommendation widgets, Figure 1 illustrates a few examples including single-product summaries serving as product titles (colored cyan) and multiple-product summaries serving as widget captions (colored orange). These widgets contain substitute recommendations and usually named *"More similar items to consider"* (or other similar and static kind of phrases). As we can see, the widget captions generated by *AmpSum* are dynamic, content-adaptive, and more explainable to the users receiving such recommendations. On the one hand, the generated widget captions summarize the similarities among recommended products, elaborating the recommendation intention (i.e., same technology, same brand). On the other hand, they highlight the main differences (i.e., key different features) suggesting users what to focus on while comparing such recommendations. It is also worth mentioning that the new product titles are now shorter than the original ones, therefore they can fit entirely into limited space of the widgets for most of the cases.

5 RELATED WORK

In this section, we review the two groups of literature most related to our problem, namely: text summarization (particularly relating to online product summarization) and explainable recommendation.

Product/Document Summarization. The vast majority of document summarization works are categorized into extractive and abstractive approaches. Various extractive methods are based on graph [7], topic model [2, 36], and supervise learning of summary/no-summary binary classification [27, 48]. Recent advances in abstractive summarization witness a success of applying neural attention network [30]. The core idea of leveraging the encoder-decoder architecture inspires [6, 26] applying recurrent network and [31] introducing pointer network to the decoders. Further improvements with neural abstractive summarization have been achieved with Transformer-based models [19, 24, 29]. Concurrently, multi-document summarization is also extensively studied with methods largely inherited from single-document summarization [2, 11, 22, 37, 41]. One direction is topic-based approach trying to identify topic representations of the document sets, which in turns are used to extract relevant sentences to form the summaries [12, 35]. Another direction is cluster-based approach where the clustering is done on the sentence graph [46] or on the sentence affinity matrix [37]. These approaches can only generate a summary as a combination of selected sentences. Such output format is undesirable and concise enough for our problem setting. As a competitive method for document summarization, BART [19] is included as a baseline. We also compare with different variants of the Pointer-Generator Networks [31] for completeness.

The line of works focusing on product summarization share some similarities to the general setting of text-based summarization while still owning its specific nature. Most works try to produce summaries based on the input of product descriptions [42] or user reviews [25, 28, 32]. Liu et al. [23] propose a neural network for review summarization leveraging both user and product information. Considering the user preferences on different product aspects, [20] introduces an encoder-decoder framework to generate aspect-oriented customized summaries, while [10] proposes to solve it with a reinforcement neighbor selection method. Xiao and Munro [40] develop extractive summarization methods for product titles by leveraging existing Named Entity Recognition (NER) systems. The methods are practical for specific use cases though not very flexible due to a fixed set of named entities. Ours is distinct from other works in this category, not only do we try to solve multiple-product summarization problem, but we also focus on the explanation aspect of the summaries that serve as titles of recommendation widgets.

Explainable Recommendation. Existing recommender systems are based on collaborative filtering [15], matrix factorization [16] and neural recommendation model [43]. Recently, to further improve user experience, great efforts have been promoted to explainable recommendation problems [17, 18, 44]. One common way to generate explanation for recommendation is to leverage knowledge graph [8, 14]. For example, Zhao et al. [47] use reinforcement learning to propose a demonstration-based knowledge graph reasoning framework for explainable recommendation, and Xian et al. [39] leverage reinforcement learning on knowledge graph to provide behavior-based explanation for product recommendations. There are also work using sentiment/opinion to perform explainable recommendation [3, 9]. Zhang et al. [45] combine sentiment analysis into factorization model to improve explainable recommendation accuracy. Wang et al. [38] develop a multi-task learning solution for explainable recommendation to jointly optimize user preference for recommendation and opinionated content generation for explanation. There are also research work around attribute-aware explainable recommendation. Hou et al. [13] extract visual attributes from product images and Chen et al. [4] develop methods to leverage both user and item attributes to generate interpretable recommendations. However, all these work mainly focus on improving recommendation accuracy yet not generating explicit and user-friendly explanation on products. Also, the explainability over multiple products as an organized widget is scarcely investigated.

6 CONCLUSION

We study the problem of multiple-product summarization in which the end goal is to arrive at concise yet informative summaries focusing on understanding how multi-product recommendations are connected to each other. We propose the AmpSum framework consisting of two modules, Pointer Transformer and Recurrent Pointer Transformer, tailored for single-product summarization and multiple-product summarization, respectively. Experiments on datasets from Amazon.com demonstrate that AmpSum consistently performs better than comparable baselines on multiple evaluation metrics. Moreover, case studies illustrate that AmpSum can adaptively generate product summaries and widget captions based on different product recommendations emphasizing practicality of the model. We also show that the generated single-product summaries can serve as compressed product title strings addressing limited space issue when displaying recommendations on the e-commerce websites. As a future direction, we are interested in exploring product summarization problem from contextual and personalized perspectives.

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] Asli Celikyilmaz and Dilek Hakkani-Tur. 2010. A hybrid hierarchical model for multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics.* 815–824.

[3] Hanxiong Chen, Xu Chen, Shaoyun Shi, and Yongfeng Zhang. 2021. Generate natural language explanations for recommendation. *arXiv preprint arXiv:2101.03392.*

[4] Tong Chen, Hongzhi Yin, Guanhua Ye, Zi Huang, Yang Wang, and Meng Wang. 2020. Try this instead: Personalized and interpretable substitute recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 891–900.

[5] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. (2014), 1724–1734.

[6] Sumit Chopra, Michael Auli, and Alexander M Rush. 2016. Abstractive sentence summarization with attentive recurrent neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies.* 93–98.

[7] Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of artificial intelligence research* 22 (2004), 457–479.

[8] Zuohui Fu, Yikun Xian, Ruoyuan Gao, Jieyu Zhao, Qiaoying Huang, Yingqiang Ge, Shuyuan Xu, Shijie Geng, Chirag Shah, Yongfeng Zhang, et al. 2020. Fairness-aware explainable recommendation over knowledge graphs. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval.* 69–78.

[9] 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.

[10] Zheng Gao, Lujun Zhao, Heng Huang, Hongsong Li, Changlong Sun, Luo Si, and Xiaozhong Liu. 2020. Behavior Based Dynamic Summarization on Product Aspects via Reinforcement Neighbour Selection. In *ECAI 2020.* IOS Press, 2022–2029.

[11] Jade Goldstein, Vibhu O Mittal, Jaime G Carbonell, and Mark Kantrowitz. 2000. Multi-document summarization by sentence extraction. In *NAACL-ANLP 2000 Workshop: Automatic Summarization.*

[12] Sanda Harabagiu and Finley Lacatusu. 2005. Topic themes for multi-document summarization. In *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval.* 202–209.

[13] Min Hou, Le Wu, Enhong Chen, Zhi Li, Vincent W. Zheng, and Qi Liu. 2019. Explainable Fashion Recommendation: A Semantic Attribute Region Guided Approach. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19.* 4681–4688.

[14] Xiaowen Huang, Quan Fang, Shengsheng Qian, Jitao Sang, Yan Li, and Changsheng Xu. 2019. Explainable interaction-driven user modeling over knowledge graph for sequential recommendation. In *Proceedings of the 27th ACM International Conference on Multimedia.* 548–556.

[15] Yehuda Koren and Robert Bell. 2015. Advances in collaborative filtering. *Recommender systems handbook* (2015), 77–118.

[16] Yehuda Koren, Robert Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42, 8 (2009), 30–37.

[17] Trung-Hoang Le and Hady W. Lauw. 2020. Synthesizing Aspect-Driven Recommendation Explanations from Reviews. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20.* 2427–2434.

[18] Trung-Hoang Le and Hady W Lauw. 2021. Explainable Recommendation with Comparative Constraints on Product Aspects. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining.* 967–975.

[19] 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. (2020), 7871–7880.

[20] Jiahui Liang, Junwei Bao, Yifan Wang, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2021. CUSTOM: Aspect-Oriented Product Summarization for EC ommerce. In *CCF International Conference on Natural Language Processing and Chinese Computing.* Springer, 124–136.

[21] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out.* 74–81.

[22] Chin-Yew Lin and Eduard Hovy. 2002. From single to multi-document summarization. In *Proceedings of the 40th annual meeting of the association for computational linguistics.* 457–464.

[23] Hui Liu and Xiaojun Wan. 2019. Neural review summarization leveraging user and product information. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management.* 2389–2392.

[24] Yang Liu and Mirella Lapata. 2019. Text Summarization with Pretrained Encoders. 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).* 3730–3740.

[25] Duy Khang Ly, Kazunari Sugiyama, Ziheng Lin, and Min-Yen Kan. 2011. Product review summarization from a deeper perspective. In *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries.* 311–314.

[26] Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning.* 280–290.

[27] You Ouyang, Wenjie Li, Sujian Li, and Qin Lu. 2011. Applying regression models to query-focused multi-document summarization. *Information Processing & Management* 47, 2 (2011), 227–237.

[28] Haojie Pan, Rongqin Yang, Xin Zhou, Rui Wang, Deng Cai, and Xiaozhong Liu. 2020. Large scale abstractive multi-review summarization (LSARS) via aspect alignment. In *SIGIR.* 2337–2346.

[29] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research* 21, 140 (2020), 1–67.

[30] Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing.* 379–389.

[31] Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).* 1073–1083.

[32] Ori Shapira and Ran Levy. 2020. Massive multi-document summarization of product reviews with weak supervision. *arXiv preprint arXiv:2007.11348* (2020).

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems.* 5998–6008.

[34] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer Networks. *Advances in Neural Information Processing Systems* 28 (2015), 2692–2700.

[35] Xiaojun Wan and Jianwu Yang. 2008. Multi-document summarization using cluster-based link analysis. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval.* 299–306.

[36] Dingding Wang, Shenghuo Zhu, Tao Li, and Yihong Gong. 2009. Multi-document summarization using sentence-based topic models. In *Proceedings of the ACL-IJCNLP 2009 conference short papers.* 297–300.

[37] Kexiang Wang, Baobao Chang, and Zhifang Sui. 2020. A Spectral Method for Unsupervised Multi-Document Summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP).* 435–445.

[38] Nan Wang, Hongning Wang, Yiling Jia, and Yue Yin. 2018. Explainable recommendation via multi-task learning in opinionated text data. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval.* 165–174.

[39] Yikun Xian, Zuohui Fu, Shan Muthukrishnan, Gerard De Melo, and Yongfeng Zhang. 2019. Reinforcement knowledge graph reasoning for explainable recommendation. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval.* 285–294.

[40] Joan Xiao and Robert Munro. 2019. Text Summarization of Product Titles. In *eCOM@ SIGIR.*

[41] Yumo Xu and Mirella Lapata. 2020. Coarse-to-fine query focused multi-document summarization. In *Proceedings of the 2020 Conference on empirical methods in natural language processing (EMNLP).* 3632–3645.

[42] Peng Yuan, Haoran Li, Song Xu, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2020. On the faithfulness for E-commerce product summarization. In *Proceedings of the 28th International Conference on Computational Linguistics.* 5712–5717.

[43] Shuai Zhang, Lina Yao, Aixin Sun, and Yi Tay. 2019. Deep learning based recommender system: A survey and new perspectives. *ACM Computing Surveys (CSUR)* 52, 1 (2019), 1–38.

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

[45] 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 *SIGIR.* 83–92.

[46] Jinming Zhao, Ming Liu, Longxiang Gao, Yuan Jin, Lan Du, He Zhao, He Zhang, and Gholamreza Haffari. 2020. SummPip: Unsupervised Multi-Document Summarization with Sentence Graph Compression. In *SIGIR.* 1949–1952.

[47] Kangzhi Zhao, Xiting Wang, Yuren Zhang, Li Zhao, Zheng Liu, Chunxiao Xing, and Xing Xie. 2020. Leveraging demonstrations for reinforcement recommendation reasoning over knowledge graphs. In *SIGIR.* 239–248.

[48] Liang Zhou and Eduard Hovy. 2003. A web-trained extraction summarization system. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics.* 284–290.

ToH:	Under Armour Women's Charged Aurora Cross Trainer (Rubber, White (100) /White, 6)
ToP:	Under Armour Women's Cross Trainer (Rubber, White (100) /White, 6)
ToP+FoH:	Under Armour Women's Charged Aurora Cross Trainer (Rubber, White (100) /White, 6)
ToH:	Diesel Men's Fastner II Fashion Sneaker (Leather, Black, 8.5)
ToP:	Diesel Men's Fashion Sneaker (Leather, Black, 8.5)
ToP+FoH:	Diesel Men's Fastner II Fashion Sneaker (Leather, Black, 8.5)
ToH:	Callaway Men's Swami Golf Shoe (Rubber, Black/Grey, 9 Wide)
ToP:	Callaway Men's Golf Shoe (Rubber, Black/Grey, 9 Wide)
ToP+FoH:	Callaway Men's Swami Golf Shoe (Rubber, Black/Grey, 9 Wide)
ToH:	ASICS Women's GEL-Cirrus33 Running Shoe (Mesh $\surd$, Black/Granite/Electric Turquoise, 10.5)
ToP:	ASICS Women's Running Shoe (Mesh, Rubber, Black/Granite/Electric Turquoise, 10.5)
ToP+FoH:	ASICS Women's GEL-Cirrus33 Running Shoe (Mesh, Rubber, Black/Granite/Electric Turquoise, 10.5)
ToH:	K-Swiss Men's Court Frasco Sneaker (Rubber, White/Bone/Marshmallow, 8)
ToP:	K-Swiss Men's Sneaker (Rubber, White/Bone/Marshmallow, 8)
ToP+FoH:	K-Swiss Men's Court Frasco Sneaker (Rubber, White/Bone/Marshmallow, 8)
ToH:	Emerica Men's The Romero Laced Skate Shoe (Lace-Up, Black/Blue, 6.5 Medium)
ToP:	Emerica Men's Skate Shoe (Synthetic, Black/Black/Blue, 6.5)
ToP+FoH:	Emerica Men's The Romero Laced Skate Shoe (Synthetic, Black/Black/Blue, 6.5)
ToH:	adidas Women's Energy Cloud w Running Shoe Three/Metallic Silver/Grey Two, 12 Medium US. /Departments/Women/Shoes/Athletic/Running/Road Running. adidas, Lace
ToP:	adidas Women's Running Shoe (Mesh-Synthetic, Rubber, Grey Three/Metallic Silver/Grey Two, 12)
ToP+FoH:	adidas Women's Energy Cloud w Running Shoe (Mesh- Synthetic, Rubber, Grey Three/Metallic Silver/Grey Two, 12)
ToH:	Skechers Sport Women's Empire Fashion Sneaker (Fabric, Charcoal/Mint, 6.5)
ToP:	Skechers Women's Fashion Sneaker (Fabric, Synthetic, Charcoal/Mint, 6.5)
ToP+FoH:	Skechers Sport Women's Empire Fashion Sneaker (Fabric, Synthetic, Charcoal/Mint, 6.5)
ToH:	Aerosoles Women's Over Drive Slip-On Loafer (Rubber, Yellow Suede, 10 Wide)
ToP:	Aerosoles Women's Slip-On Loafer (Rubber, Yellow Suede, 10 Wide)
ToP+FoH:	Aerosoles Women's Over Drive Slip-On Loafer (Rubber, Yellow Suede, 10 Wide)
ToH:	Cole Haan Men's Nantucket Loafer (Acorn Leather, 11, 11)
ToP:	Cole Haan Men's Loafer (Leather, Rubber, Acorn Leather, 11)
ToP+FoH:	Cole Haan Men's Nantucket Loafer (Leather, Rubber, Acorn Leather, 11)

Figure 8: Examples of single-product summaries produced by *AmpSum* with three different training approaches: Train on Human Labels (*ToH*), Train on Pseudo Labels (*ToP*), and Train on Human Labels + Fine-tune on Human Labels (*ToP+FoH*). Summaries by *ToP+FoH* contain important descriptive keywords (colored green) that are missed by *ToP*. There are also duplicate/wrong phrases (colored yellow) by *ToH*. These examples are on the test set and have not been observed by the models.

7 APPENDIX

In Figure 8, we showcase several examples of single-product summaries produced by *AmpSum* with three different training approaches: Train on Human Labels (*ToH*), Train on Pseudo Labels (*ToP*), and Train on Human Labels + Fine-tune on Human Labels (*ToP+FoH*). The quantitative performances of these three approaches are previously reported in Figure 5. For these examples, we look at cases that the generated summaries by *ToP+FoH* can perfectly match the human labels in order to observe the effectiveness of the fine-tuning step. After being fine-tuned on human labels, *ToP+FoH* could generate important descriptive keywords (highlighted in green) that were missed by *ToP*, which is only trained on pseudo labels. There are cases that *ToH* could generate good summaries, though there are also bad cases with wrong focuses and duplicates (highlighted in yellow). Noted that these examples are on the human evaluation set and have not been observed by the models during training and fine-tuning phases.

Figure 9 presents examples of multiple-product summaries produced by *AmpSum* and *HBART*, together with *Human Label*. Here we would like to seek qualitative understanding of *AmpSum*'s better performance of as compared to *HBART* (Section 4.3). Overall, we observe that the summaries generated by *AmpSum* are closer to *Human Label* as compared to the ones generated by *HBART*. For the summaries by *HBART*, we notice false claims and duplicates (highlighted in yellow), while there are more agreements (highlighted in green) between summaries by *AmpSum* and *Human Label*. Those important agreements are also missed from the summaries by *HBART* in some cases. Similarly, these demonstrated examples are on the human evaluation set and have not been observed by the models during training and fine-tuning phases.

HBART+Perm:	15.6 Inch DDR4 Ram
AmpSum (GRU):	15.6 Inch Gaming Laptops, Different Brands and Cpu Manufacturers
Human Label:	Gaming Laptops with 15.6 FHD Display Technology, Different Brands and Cpu Manufacturers
HBART+Perm:	Blooming Blooming Blooming For Epson Projectors For Epscors
AmpSum (GRU):	Epson Projectors with Different Controllers
Human Label:	Generic Remote Controllers For Epson Projectors
HBART+Perm:	Dell Inspiron Laptops with Different Hard Disk Sizes
AmpSum (GRU):	Dell Inspiron Laptops with Touchscreen Technology, Different Display Sizes
Human Label:	Dell Inspiron with FHD Touchscreen Technology, Different Display Sizes, Colors and Hard Disk Sizes
HBART+Perm:	Laptops with 15.6 Display Sizes and Memory Storage
AmpSum (GRU):	Laptops with 15.6 Display Sizes, Different Models
Human Label:	Hidevolution Laptops with 15.6 Inch Screens and Windows OS with Different Models
HBART+Perm:	Tablets with Different Brands, Model Names, Colors, Colors
AmpSum (GRU):	Tablets with Different Brands, Display Sizes
Human Label:	Tablets with Different Brands, Display Sizes and Memory Storage Capacity
HBART+Perm:	TVs with Different Brands, Display Sizes and Connectivity Technology
AmpSum (GRU):	TVs with LED Technology, Different Brands and Display Sizes
Human Label:	LED Display TVs with Different Brands, Display Sizes and Supported Internet Services
HBART+Perm:	ASUS with Different Brands, Colors, Display Sizes
AmpSum (GRU):	15 Inch Laptops with ASUS Brand, Different Memory Sizes
Human Label:	ASUS 15 Inch Laptops with Windows OS, Different Models, Display Sizes and Memory
HBART+Perm:	Projectors with Different Brands, Hardware
AmpSum (GRU):	Digital Projectors with Different Brands, Brightness and Display Resolution
Human Label:	Digital Projectors with 1080P Support, Different Brands, Display Resolution, Hardware Interfaces and Brightness
HBART+Perm:	Laptops with LG Display with 4K Display with Different Brands, Display Sizes
AmpSum (GRU):	LG Smart TVs with Alexa Technology, Different Display Sizes
Human Label:	4K LG Smart TVs with Alexa Built-In, Different Display Sizes
HBART+Perm:	Fujifilm FinePix Digital Cameras with Different Colors and Display Sizes
AmpSum (GRU):	Fujifilm Digital Cameras with Different Colors and Lens Types
Human Label:	Fujifilm Digital Cameras with Different Models, Colors and Included Accessories

Figure 9: Examples of multiple-product summaries produced by *AmpSum* and *HBART*, together with *Human Label*. Overall, summaries generated by *AmpSum* are closer to *Human Label* as compared to the ones by *HBART*. There are false claims and duplicates (highlighted in yellow) by *HBART*, and there are agreements (highlighted in green) between *AmpSum* and *Human Label* that are also missed by *HBART*. These examples are on the test set and have not been observed by the models.