Automatic Product Copywriting for E-commerce

Xueying Zhang[1*], Yanyan Zou[2*], Hainan Zhang[2], Jing Zhou[2], Shiliang Diao[2], Jiajia Chen[2], Zhuoye Ding[2], Zhen He[2], Xueqi He[2], Yun Xiao[1], Bo Long[2], Han Yu[3†], Lingfei Wu[1†]

[1] JD.COM Silicon Valley Research Center
[2] JD.COM
[3] Nanyang Technological University
han.yu@ntu.edu.sg, lingfei.wu@jd.com

Abstract

Product copywriting is a critical component of e-commerce recommendation platforms. It aims to attract users' interest and improve user experience by highlighting product characteristics with textual descriptions. In this paper, we report our experience deploying the proposed Automatic Product Copywriting Generation (APCG) system into the JD.com e-commerce product recommendation platform. It consists of two main components: 1) natural language generation, which is built from a transformer-pointer network and a pre-trained sequence-to-sequence model based on millions of training data from our in-house platform; and 2) copywriting quality control, which is based on both automatic evaluation and human screening. For selected domains, the models are trained and updated daily with the updated training data. In addition, the model is also used as a real-time writing assistant tool on our live broadcast platform. The APCG system has been deployed in JD.com since Feb 2021. By Sep 2021, it has generated 2.53 million product descriptions, and improved the overall averaged click-through rate (CTR) and the Conversion Rate (CVR) by 4.22% and 3.61%, compared to baselines, respectively on a year-on-year basis. The accumulated Gross Merchandise Volume (GMV) made by our system is improved by 213.42%, compared to the number in Feb 2021.

Introduction

With the rapid development of the Internet, online shopping has become indispensable in people's daily life. E-commerce is now a mainstay way of shopping for many people all over the world. Product advertising copywriting plays an important role in e-commerce platforms, where well-written advertisements can help attract customers and, in turn, increase sales. Traditionally, product copywriting is performed by human copywriters. This approach has important limitations. Firstly, the efficiency of human copywriters cannot match the growth rate of new products. Secondly, manpower cost increases as the number of products in the system increases. Last but not least, specialist training and tutoring is required for copywriters for different products and scenarios. To overcome these issues, automatic product copywriting generation has become an important line of research in e-commerce.

Natural language generation (i.e., NLG) research focuses on the construction of computer systems that can produce understandable texts in human languages from some underlying non-linguistic or textual representation of information (Reiter and Dale 2000). NLG systems combine knowledge from both the language and the application domain to automatically generate desired texts. In recent years, NLG techniques have been developing rapidly and applied to a wide range of applications. For instance, abstract text summarization (Rush, Chopra, and Weston 2015; Banko, Mittal, and Witbrock 2000; Knight and Marcu 2000) generates a summarized version of input text to highlighted certain information; dialogue generation (Tao et al. 2018; Zhang et al. 2020) generates responses according to the previous chats; machine translation (Bahdanau, Cho, and Bengio 2016) translates the input texts into a different language; and storytelling or poem generation (Yao et al. 2019; Li et al. 2018) generates creative texts.

In the early stage of text generation research, due to limitation of data and computational resources, models relied heavily on feature engineering. In this case, features are extracted from raw data based on human knowledge and preference, and fed into the models to learn from the limited data (Lafferty, McCallum, and Pereira 2001; Och et al. 2004). With the development of deep learning, features are learned automatically and defined implicitly during model training. The natural language generation problem has been framed as a sequence-to-sequence task and various studies have been proposed to enhance the model architecture, such as convolutional neural networks (CNN) (Gehring et al. 2017), recurrent neural networks (RNN) (Sutskever, Vinyals, and Le 2014), graph neural networks (GNN) (Wu et al. 2020), and attention-based architecture (Bahdanau, Cho, and Bengio 2016; Vaswani et al. 2017) (especially the transformer network (Vaswani et al. 2017)).

Recently, the pre-training plus fine-tuning paradigm has gained traction and been widely applied in real-world applications. Under this paradigm, models are first pre-trained on large-scale corpus (Devlin et al. 2019a; Brown et al. 2020), or on domain-specific corpus (Zhang et al. 2021) with well-defined pre-training tasks. The resulting pre-trained models are then fine-tuned to adapt to different downstream tasks. Research in this paradigm mainly focuses on defining training objectives in the pre-training and fine-tuning stages.

[*] These authors contributed equally.

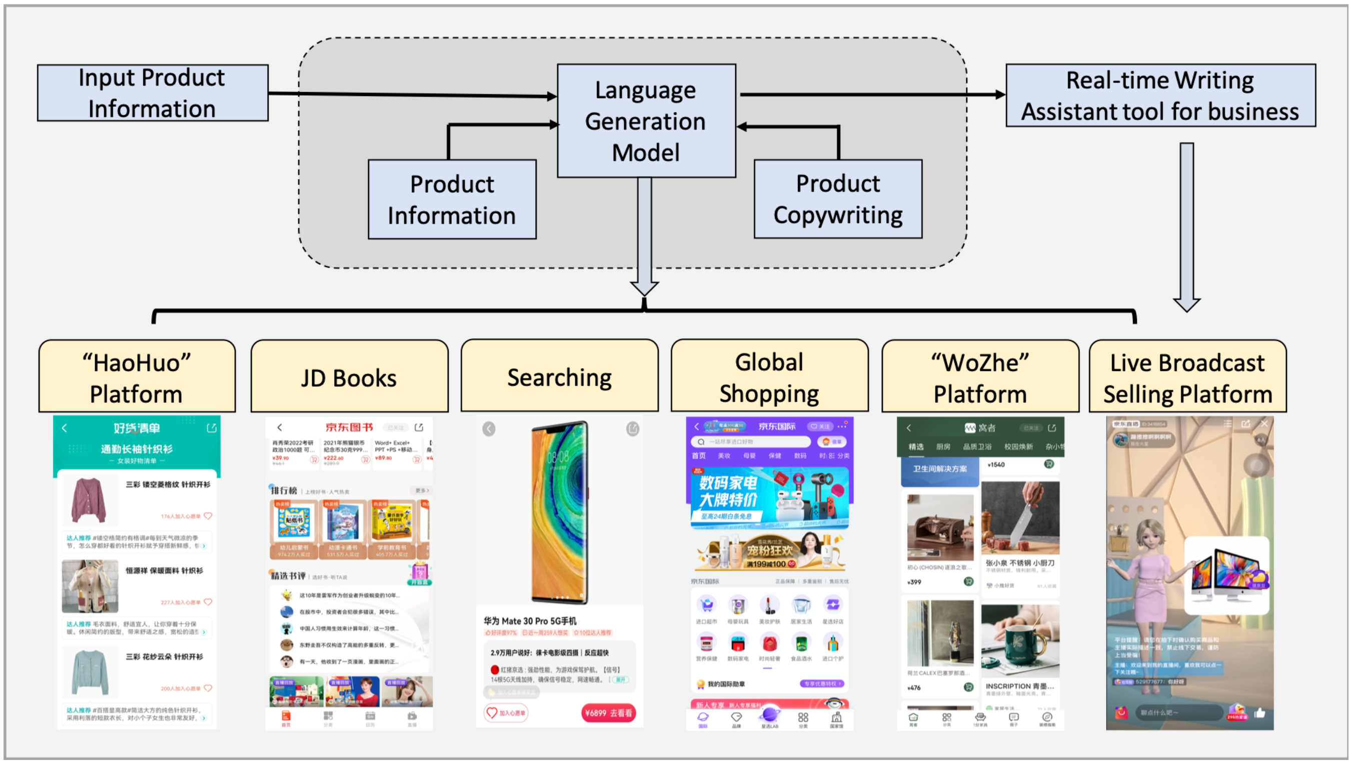

Input Product Information
Language Generation Model
Real-time Writing Assistant tool for business
Product Information
Product Copywriting
"HaoHuo" Platform
JD Books
Searching
Global Shopping
"WoZhe" Platform
Live Broadcast Selling Platform

Figure 1: Overview of the proposed Automatic Product Copywriting Generation (APCG) system.

With increasing computational resources and the emerging transformer architecture (Vaswani et al. 2017), recent studies (e.g., GPT2 (Radford et al. 2019), CTRL (Keskar et al. 2019), UNiLM (Dong et al. 2019)) utilizing pre-training and fine-tuning techniques have achieved outstanding performance in language generation tasks.

In this paper, we report on our experience deploying the proposed Automatic Product Copywriting Generation (APCG) system into the JD.com[1] e-commerce product recommendation platform. It is made up of two main components: 1) a natural language generation (NLG) module, which is built from a transformer-pointer network and a pre-trained sequence-to-sequence model; and 2) a copywriting quality control module, which involves both automatic evaluation and human screening to ensure the quality of the generated descriptions. For selected domains, the models are trained and updated daily with the updated training data. In addition, the model is also used as a real-time writing assistant tool on our live broadcast platform.

Since its deployment in JD.com in Feb 2021, APCG has generated 2.53 million product descriptions by Sep 2021. As a result, the overall click-through rate (CTR) has been improved by 4.22%, the conversion rate (CVR) has been improved by 3.61%, and the accumulated gross merchandise volume (GMV) has more than doubled on a year-on-year basis.

Application Description

An online e-commerce website typically describes a product p from several aspects, including the product title t, a set of the product attribute A and the corresponding attribute values V for each attribute $a \in A$, as well as the short advertisement slogan s written by advertising experts. The product title t usually describes the product with a short text, expressing the key information, such as the brand name and product type. The product attribute set A captures the properties of a product from different perspectives. For example, the product functions, intended users, styles, colors and materials. The corresponding values of the attributes from A form the product attribute value set V. For example, the product function feature can be *breathable, resistant*, and the intended user feature can be *ladies*. Given the title t, attribute set A, the corresponding attribute value set V and the advertisement slogan s, the task of product copywriting is to generate a well-written description D that is able to present product characteristics for the e-commerce system to attract user interest and keep them informed of the key features of the product quickly. An example of product copywriting is demonstrated in Figure 4.

Early work on product copywriting applied template-based generation methods with statistical knowledge extracted from the training data (Wang et al. 2017), while the template coverage and diversity are limited by the training corpus. Chen et al. 2019 designed a transformer-based generation model to incorporate the external knowledge (i.e., Wikipedia knowledge base). Besides, the textual informa-

[1] https://www.jd.com/

tion (i.e., product title and attributes) and the product image features are also incorporated. The performance of such a model heavily relies on the availability of related knowledge. Similarly, Li et al. 2020 constructed a set of aspects (i.e., salient attributes) and the corresponding keywords for each aspect, which, however, requires expert annotations. An adaptive posterior Transformer-based network (Zhan et al. 2021) has been designed to utilize relevant information from customer reviews extracted by an adaptive posterior distillation module, where the product title and attribute information is considered as the main model input.

However, neither the external knowledge base nor the customer reviews are always accessible. In practice, for the e-commerce platform, newly released products are emerging every day, where the product information is quite limited. In addition, such kind of external features may contain noise, especially for newly released products. This reduces the quality of the generated product descriptions. On the other hand, each product is accompanied with several short advertising phrases written by experts. However, existing studies did not leverage this information when generating product descriptions. In this work, we jointly consider the product textual features (i.e., title, attribute and advertising slogan) as product input information, and develop a transformer-pointer model and a pre-trained sequence-to-sequence model for automatic product copywriting in e-commerce settings. Both of such two models constitute our Automatic Product Copywriting Generation (APCG) system.

As illustrated in Figure 1, our proposed APCG system is designed to serve a variety of applications in the JD.com e-commerce platform, including the "HaoHuo" (referring to "Discovery Goods") product recommendation platform, JD books, searching, global shopping, the "WoZhe" platform (mainly for home appliance) and the live broadcast selling platform, etc. As shown in Figure 2, the overall system consists of the following steps:

1) NLG model training: To collect data, including both product information and descriptions, for training the NLG model. Different models are selected upon different scenarios. Both the transformer-pointer network and the pre-trained language models are applied in this stage.
2) Post-evaluation for direct customer use: Post-processing models are utilized here to filter out contents which do not meet the requirements. Human screening is performed after model-based filtering. The contents produced here are directly used by the product recommendation platform for JD customers.
3) Real-time writing assistant tool for sellers: Besides generating and filtering contents based on proposed product skus, a real-time writing assistant tool based on APCG has been developed for JD sellers. With this tool, JD sellers can input their desired products, and it will automatically generate product descriptions. Sellers can then edit the generated contents, and display them to customers.
4) Automatic procedure with daily update: APCG is trained daily to meet the latest writing preference based newly updated data.

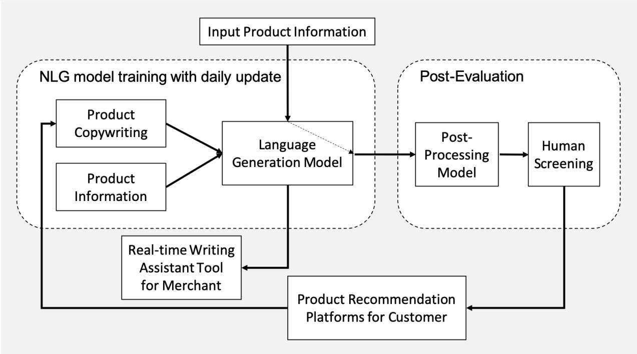

Input Product Information
NLG model training with daily update
Post-Evaluation
Product Copywriting
Product Information
Language Generation Model
Post-Processing Model
Human Screening
Real-time Writing Assistant Tool for Merchant
Product Recommendation Platforms for Customer

Figure 2: Workflow of the APCG system.

Use of AI Technology

In this section, we describe the key techniques adopted by the APCG AI Engine.

Data Collection and Cleaning

Data Collection We built a large-scale real-world dataset from our in-house platform which includes two parts: 1) product description data, and 2) product information data. The product descriptions are written by professional copywriters with good knowledge of marketing. For each product, the description includes both advertising contents and the title. For the product information data, we collected multiple types of data from our in-house database, including product titles, attributes, product detail images and user reviews.

Data Cleaning After obtaining the large-scale datasets, we perform filtering and cleaning on the raw data to construct our training dataset. The product descriptions are filtered by human evaluation and rule-based methods, and used as the outputs of our model. The inputs include information from product titles, attributes, product detail images and user reviews. For each category, we select different groups of attributes which have high relevance with the target products. For product detail images, we leverage the optical character recognition (OCR) and classification techniques to extract key information about the product. Firstly, texts are extracted from the product detail images. Then, the extracted texts are ranked in descending order of their importance and relevance using the classification model. Finally, highly ranked texts are selected and merged as the final OCR input. We use the ranking model to select useful reviews, and a summarization model to obtain highlights of selected reviews, which are then used as additional inputs.

The Transformer-Pointer Network

Based on this large-scale proprietary dataset, the automatic product copywriting problem can be regarded as an abstractive summarization problem. We designed a pointer-generator network, which allows the transformer network to generate new words and copy words from the input text at

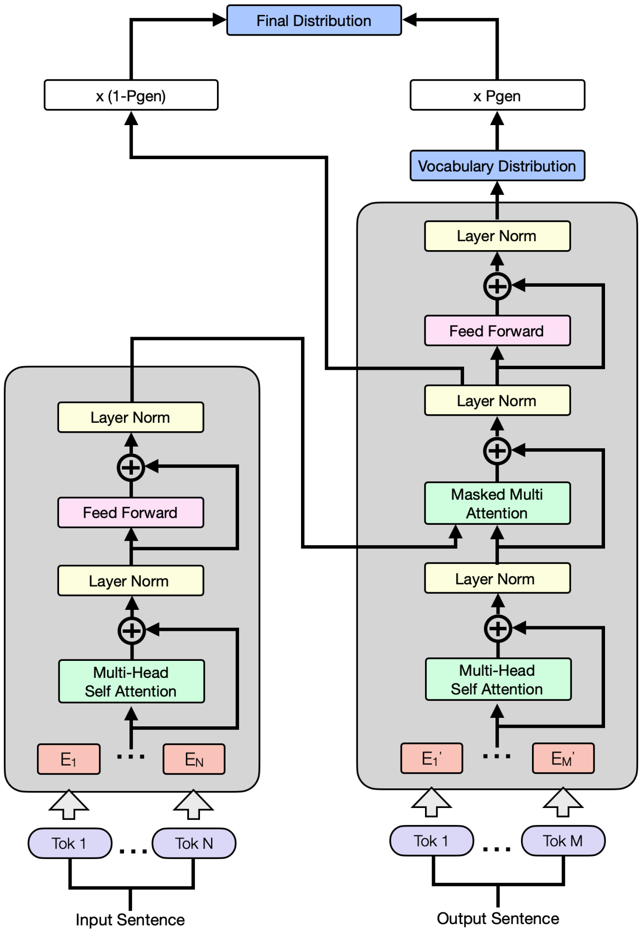


Figure 3: The transformer-pointer network structure.

the same time. Through this transformer-pointer network, issues of inaccurate descriptions and unknown words can be handled since both extractive and abstractive approaches are executed simultaneously.

The proposed transformer-pointer network is a combination of a pointer-generator network and a transformer structure. As shown in Figure 3, by introducing the trainable probability P_{gen}, the model learns to choose whether to use words from the vocabulary source or from the input text. The model will select a new word from the vocabulary distributions with a probability of P_{gen}, and copy words directly from the input text through the selective pointer with a probability of $(1-P_{gen})$. We take the encoder-decoder attentions in the last decoder layer as the copy distribution. Note that for the multi-head attention, we obtain the copy distributions by summing across multiple heads.

The Pre-trained Language Model

Training a neural network from scratch requires large amounts of data with high-quality labels, especially true for the generation task. Unlabeled open-domain data (i.e., without gold standard annotations) are readily available from online sources such as Wikipedia. Recently, pre-trained models, such as OpenAI GPT (Radford et al. 2018, 2019), BERT (Devlin et al. 2019b), RoBERTa (Liu et al. 2019) and BART (Lewis et al. 2019), have become popular for NLP tasks. They have significantly improved the state-of-the-art performance on various natural language understanding (NLU) tasks (e.g., SQuAD question answering (Rajpurkar et al. 2016), natural language inference (Bowman et al. 2015), text classification (Zhang, Zhao, and LeCun 2015)) and natural language generation tasks (e.g., machine translation (Song et al. 2019) and text summarization (Zou et al. 2020)). This demonstrates the effectiveness and transferability of models pre-trained on large scale unlabeled data. Recent research progresses in (Brown et al. 2020) have demonstrated that pre-trained models can play as an important role in few-shot learning with limited training instances.

In the product copywriting scenario, there are different categories of products (e.g., clothing, mobile phones, sportswear). The product characteristics and data properties vary significantly across different categories. One straightforward solution is to train the transformer-pointer model on the training data for each category individually. However, this will incur high computation and maintenance costs. In addition, as the products in the e-commerce are updated frequently, there is often limited or even no training data for newly released products in practice. Therefore, we adopt the pre-trained model for product copywriting.

We adopt the sequence-to-sequence transformer network (Vaswani et al. 2017), which comprises of an encoder and a decoder, as the backbone architecture. Given an input token sequence $X = (x_1, x_2, \ldots, x_{|X|})$ paired with its corresponding output sequence $Y = (y_1, y_2, \ldots, y_{|Y|})$, we aim to learn the model parameters θ and estimate the conditional probability:

$$P(Y|X;\theta) = \prod_{t=1}^{|Y|} p(y_t|y_{<t};X;\theta) \qquad (1)$$

where $y_{<t}$ stands for all tokens before position t (i.e., $y_{<t} = (y_1, y_2, \ldots, y_{t-1})$). Given the whole training set $(\mathcal{X}, \mathcal{Y})$, this model can be trained by maximizing the log-likelihood of the training input-output pairs:

$$\mathcal{L}(\theta;\mathcal{X},\mathcal{Y}) = \sum_{(X,Y)\in(\mathcal{X},\mathcal{Y})} \log P(Y|X;\theta). \qquad (2)$$

We pre-train the model using sequence-to-sequence objectives.

Recall that the attributes of a product are represented by the set A without ordering information. We observe that the transformer model generates different outputs when receiving attributes in different orders as inputs. This can sometimes produce poor quality outputs. On the other hand, given the product title, attributes and advertising slogans, the task of product copywriting is to generate the description, where the generated text contains tokens copied from input text and novel words and phrases that are not contained in the input. Based on these observations, we design two sequence-to-sequence objectives for pre-training.

Sentence Re-ordering Inspired by (Zou et al. 2020), we incorporate the sentence re-ordering objective (SR) in APCG. We first divide an unlabeled document into multiple sentences based on full stop signs. For better demonstration, we change the notation of a document slightly in this paragraph. Let $X = (S_1||S_2||\dots||S_m)$ denote a document, consisting of several sentences, where S_i is a sentence, m is the number of sentences, and $||$ refers to sentence concatenation. The sentence index order in X can be represented as $\mathcal{O} = (1, 2, \dots, m)$. We then shuffle the sentences in the document. In other words, the items in the order $\mathcal{O}$ are re-arranged and we obtain a shuffled order $\mathcal{O}_S = (a_1, a_2, \dots, a_m)$, where a_i refers to sentence index, $1 \le a_i \le m$, $1 \le a_j \le m$, and $a_i \ne a_j$ for any $i, j \in [1, m]$ and $i \ne j$. Concatenating sentences following $\mathcal{O}_S$, we obtain a *shuffled* document $\hat{X}_S = (S_{a_1}||S_{a_2}||\dots||S_{a_m})$. The sequence-to-sequence model takes the shuffled document $\hat{X}_S$ as input and is pre-trained to re-instate the original one X. The training objective is expressed as:

$$\mathcal{L}(\theta; \mathcal{X}) = \sum_{X \in \mathcal{X}} \log P(X|\hat{X}_S; \theta).$$

Pseudo Summary Generation This objective is adopted from (Su 2021). Suppose a document is denoted as $X = (S_1||S_2||\dots||S_m)$. We select around $m/4$ sentences from the document X, and concatenate into $\hat{X}_M$. The remaining $3m/4$ sentences are concatenated into X_R. The sentences in $\hat{X}_M$ are carefully selected so that $\hat{X}_M$ and X_R contain the longest subsequences among all alternatives. The longer text X_R is considered as the model input, while the shorter text $\hat{X}_M$ is regarded as the model output. Thus, the training objective is expressed as:

$$\mathcal{L}(\theta; \mathcal{X}) = \sum_{X \in \mathcal{X}} \log P(X_R|\hat{X}_M; \theta).$$

After a sequence-to-sequence model is pre-trained, we fine tune the model on the parallel copywriting training data, consisting of instances from all categories.

Post-Processing

Although the proposed language generation models can generate texts with good quality most of the time, there is still possibility that the generated descriptions might not be accurate. Thus, we incorporate post-processing models into APCG to further enhance copywriting quality to meet the high expectation in practical e-commerce applications.

In some cases, the descriptions generated by the model are not consistent with product information. For example, for iPhone 11, the model generates the product description of “iPhone 11采用升降式摄像头” (iPhone 11 with a foldable webcam), which is not an actual feature of iPhone 11. Although cases like this are rare, they can negatively affect user experience. To overcome this issue, we designed a rule-based product-word and number checking model. We collect words and numbers related for each product item, and construct our specific e-commerce domain specific word dictionary. We develop rule-based models for each product category based on these dictionaries to filter out generated descriptions with inaccurate information. The rules are based on combinations of terms, where single words within them are not affected.

Although NLG models are trained to achieve good fluency and grammar correctness, they are not fool proof, especially when the input information is too new for the model. To address this limitation, we train grammar checking models and leverage the Adaboost algorithm to deploy these models to filter out generated descriptions with poor grammar quality. These post-processing approaches help APCG filter out poorly generated description and enhance the copywriting quality.

Application Development and Deployment

Offline Evaluation

We compare our transformer-pointer model (TP) and pretraining based copywriting generation model (PCG) with two baselines before making deployment decisions:

- The template-based generation model (i.e., Template) (Wang et al. 2017); and
- The attribute mining based generation model (i.e., A-Mining) (Li et al. 2020).

For fair comparison, we randomly selected 500 products from different categories, and generated descriptions using the four models. We used SacreBLEU (Post 2018), ROUGE (Lin 2004), BLEU (Papineni et al. 2002), and Meteor (Lavie, Sagae, and Jayaraman 2004) to measure the quality of different generation model outputs. We report the SacreBLUE, ROUGE-1, ROUGE-2, ROUGE-L, BLEU-1, BLEU-2, BLUE-3, BLEU-4 and Meteor scores in Table **??**. The larger the values of these metrics, the better the performance of a model. It can be observed that PCG consistently achieves the best performance under all evaluation metrics.

Since descriptions generated by models may produce incoherent or grammatically incorrect outputs, we also evaluated these four models by eliciting human judgements. Ten human participants are presented with a product and a list of descriptions generated by the four different models in random orders. Then, they are asked to rank these descriptions according to fluency (*is the description grammatically sound?*), relevance (*is the description relevant to the product?*), and diversity (*does the text describe the product from different perspectives?*) The results are listed in Table **??**. It can be observed that PCG achieves the best performance under all human evaluation metrics. These results helped us make the decision to incorporate TP and PCG into the APCG system.

Deployment Usage

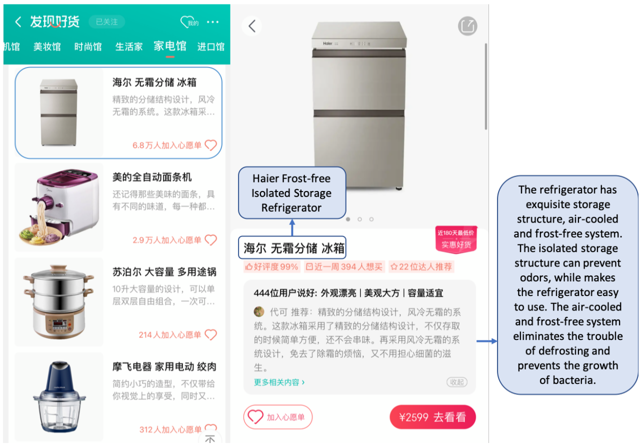
The procedure of automatic product copywriting generation includes the following steps: 1) obtaining the required product skus from the database; 2) generating product descriptions and performing post-processing; 3) submitting the generated contents for human screening; 4) storing approved contents into the database for customer service provision. Daily uploads to the descriptions database can reach up to 28,570 generated descriptions, with an average daily acceptance rate of 80%. The usage of the JD product recommen-

Model	SacreBLUE	ROUGE-1	ROUGE-2	ROUGE-L	BLEU-1	BLEU-2	BLEU-3	BLEU-4	Meteor
Template	4.84	14.20	2.65	11.01	27.95	15.94	10.09	6.80	12.72
A-Mining	5.35	16.33	3.43	13.04	30.24	17.31	10.83	7.27	13.57
TP	5.81	17.37	4.14	14.06	29.80	17.90	11.70	**8.17**	13.77
PCG	**6.19**	**18.98**	**4.37**	**15.66**	**30.29**	**18.34**	**11.97**	**8.17**	**14.04**

Table 1: Offline evaluation results for the four models.

Model	Fluency	Relevance	Diversity
Template	1.78	1.89	1.79
A-Mining	1.80	1.90	1.77
TP	1.80	1.84	1.81
PCG	**1.96**	**1.91**	**1.87**

Table 2: Human evaluation results for the four models.



海尔 无霜分储 冰箱
精致的分储结构设计，风冷无霜的系统。这款冰箱采…
6.8万人加入心愿单

美的全自动面条机
还记得那些美味的面条，具有不同的味道，每一种都…
2.9万人加入心愿单

苏泊尔 大容量 多用途锅
10升大容量的设计，可以单层双层自由组合，一次可…
214 人加入心愿单

摩飞电器 家用电动 绞肉
简约小巧的造型，不仅带给你视觉上的享受，同时又…
312 人加入心愿单

Haier Frost-free Isolated Storage Refrigerator

海尔 无霜分储 冰箱

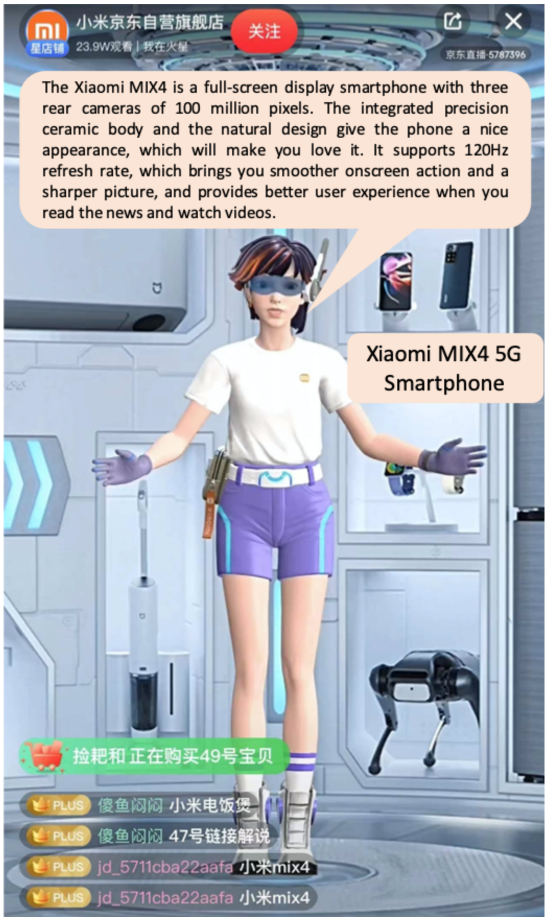
444位用户说好: 外观漂亮 | 美观大方 | 容量适宜

代可 推荐：精致的分储结构设计，风冷无霜的系统。这款冰箱采用了精致的分储结构设计，不仅存取的时候简单方便，还不会串味。再采用风冷无霜的系统设计，免去了除霜的烦恼，又不用担心细菌的滋生。

The refrigerator has exquisite storage structure, air-cooled and frost-free system. The isolated storage structure can prevent odors, while makes the refrigerator easy to use. The air-cooled and frost-free system eliminates the trouble of defrosting and prevents the growth of bacteria.

Figure 4: Examples of APCG generated descriptions on the JD e-commerce product recommendation platform.

dation platform generates new data daily, our NLG models are trained and updated daily with these new data.



The Xiaomi MIX4 is a full-screen display smartphone with three rear cameras of 100 million pixels. The integrated precision ceramic body and the natural design give the phone a nice appearance, which will make you love it. It supports 120Hz refresh rate, which brings you smoother onscreen action and a sharper picture, and provides better user experience when you read the news and watch videos.

捡耙和 正在购买49号宝贝

PLUS 傻鱼闷闷 小米电饭煲

PLUS 傻鱼闷闷 47号链接解说

PLUS jd_5711cba22aafa 小米mix4

PLUS jd_5711cba22aafa 小米mix4

Figure 5: Examples of APCG generated descriptions on the JD live broadcast selling platform. The virtual salesperson reads the product descriptions in the virtual studio.

As illustrated in Figure 2, the deployed APCG system serves two main usage scenarios: 1) content generation for customers, and 2) real-time writing assistant for sellers. In the first scenario, the contents generated by APCG are stored in our database and displayed directly on the JD product recommendation platform. Figure 4 shows an example of the product recommendation platform for customers, "Hao-Huo". The English translations of the generated texts are shown in the pop up bubbles in the figure. The generated product descriptions, paired with a short title, is pushed to the Discovery Goods Channel in the JD e-commerce website. The short title consists of three phrases: 1) the brand name, 2) one of salient product characteristics, and 3) the product name, which make up a product headline. To further ensure the user experience, we also display three images for a product. Overall, the short title, description and three images are combined together to depict a product in the Discovery Goods Channel. APCG only focuses on the description generation task.

In the second scenario, a real-time writing assistant tool is designed for sellers, especially for sellers promoting products on the live broadcast platform. The sellers can input the product ID they would like to advertise, and the APCG writing assistant tool will generate product descriptions fitting for the current context in real time. Figure 5 shows the use case in the live broadcast selling platform. By Sep, 2021, the APCG writing assistant tool has been adopted by more than 2,000 merchants with daily live streams up to 150 sessions.

Service	QPS	avg latency (ms)	TP-99 (ms)
CPU predictor	23.0	80.0	106.0
GPU predictor	100.0	15.6	19.2

Table 3: Latency test comparison of CPU and GPU services.

As real-time performance is important for the second use case, we optimize the design of APCG for this purpose. We split the encoder and decoder of our transformer model, and design interfaces to realize the beam search algorithm. The two main interfaces include the encoder predictor and decoder predictor. The encoder predictor receives the input tokens, and returns the token IDs and the encoded embedding. For the decoder predictor, the inputs include the token IDs, the encoded embedding and the current prefix. It outputs the top-k candidates of the predicted indices, tokens and probabilities. Due to language generation problems, contextual words need to be generated one-by-one. Thus, when generating descriptions for one item, the encoder interface is called once and the decoder interface is called multiple times depending on the length of the context and the beam size. For beam search, extra storage is required for holding the candidates within the beam.

We conduct latency test for our real-time writing assistant tool. Table **??** shows that the GPU predictor can handle high query request traffic of up to 100 queries per second (QPS), which is four times the throughput of the CPU predictor. The average latency and TP-99 (i.e., the minimum time under which 99% of requests have been served) achieved by using GPU are about 1/5 of using CPU.

To further enhance the efficiency, before calling our language generation engine to generate contents in the real time, we first check whether the proposed input product is in our copywriting description database. If so, we will directly provide the previously generated contents stored in the database to save time.

Maintenance

The key performance indicator used to monitor the performance of APCG is the daily acceptance rate. If this rate becomes low, we will perform manual checks on the rejected descriptions and fix the issues in the model accordingly. Besides this, no other major maintenance task has been required since its deployment in Feb 2021.

Application Use and Payoff

The APCG system has been deployed in JD.com since Feb 2021. To demonstrate the payoff generated by APCG, we compare key performance indicators before and after deploying APCG on the JD Discover Goods Channel and its mobile app. A standard A/B testing is used on such online platforms to evaluate the benefit of the APCG-generated product descriptions for the business on the platform. The impact on the business of the platform is measure by the Click-Through Rate (CTR) and Conversion Rate (CVR). CTR is defined as:

$$\text{CTR} = \frac{\#\text{clicked\_product}}{\#\text{pv\_product}} \tag{3}$$

where click_product is the number of clicks on the products, and pv_product is the number of page views for the product shown to the users. CVR is defined as:

$$\text{CVR} = \frac{\#\text{converted\_product}}{\#\text{clicked\_product}} \tag{4}$$

where the converted_product refers to the number of times a product has been purchased by users.

For online deployment, we randomly selected around 500,000 products covering 20 categories, and generated corresponding descriptions using both the Transformer-Pointer and Pre-trained models. The baseline method is the A-Mining model which was used by the platform prior to the deployment of APCG. We first conduct A/B testing for the baseline model and Transformer-Pointer model. During the online A/B testing period, over five million people visited the Discovery Good Channel, providing us with adequate opportunities for testing. The A/B testing system divided these visiting users into two equal-sized groups, and directed them into two individual buckets (i.e., buckets A and B). For users in bucket A (i.e., the baseline group), the system shows them the product descriptions generated by the A-Mining model. For users in bucket B (i.e., the experiment group), the system shows them the product descriptions generated by the Transformer-Pointer model in APCG.

The CTR and CVR for all products during the A/B testing period are calculated. We found that the our Transformer-Pointer model significantly outperforms the previous method by a large margin. Specifically, the TP model improved the CTR by 3.44% and CVR by 3.40% over the A-Mining model. The increase of the CTR indicates that users prefer to click the products with descriptions generated by our system. The improvement of CVR demonstrates that the descriptions generated by our model is successful in convincing users to make purchases.

Another A/B test has been conduced with our pre-trained model (PCG). This time, we use our Transformer-Pointer model as the baseline. All other A/B test settings remain the same. Compared to the baseline, the CTR and CVR are improved by a relatively small margin of 1.3% and 0.8%, respectively, which is in line with expectation. Recall that, the Transformer-Pointer model is deployed individually for each category, while the PCG model is able to provide description generation services for all categories. However, the pre-trained model incurs high computation costs. Thus, during APCG deployment, PCG mainly focuses on categories with limited training data or newly released products.

Between Feb 2021 and Sep 2021, APCG has generated 2.53 million product descriptions for the JD.com e-commerce platform. The overall averaged CTR achieved by APCG is 4.22% higher than that achieved by the previous system year-on-year (yoy). The improvement in terms of CTR is 3.61% yoy. The accumulated Gross Merchandise Volume (GMV) made by our system is improved by 213.42%, compared to the start number in Feb 2021. By introducing automated copywriting, our daily uploads to the descriptions database can reach up to 28,570 generated descriptions, equivalent to the productivity of thousands of human experts, which significantly saves human labor costs. Although the product copywriting task in the JD Discover Goods Channel is currently jointly performed by expert writers and the APCG systems, we observed that expert writers often prefer to create descriptions for popular products. Differently, our system (especially the pre-trained model) is able to cover both popular and long-tail products, which helps build a healthy ecosystem. Thus, it is expected that the AI system will take over this task completely in the near future.

Lessons Learned During Deployment

At the time of submission of this paper, APCG has been successfully deployed for around six months and has led to significant positive impact on JD.com's business. Several lessons we have learned during model deployment could be beneficial for other like-minded researchers and practitioners who wish to deploy cutting-edge AI technologies into real-world applications.

Firstly, besides the model capacity, the quality of training data is of paramount importance. The cleaning procedures of raw data (e.g., removing poor samples from training set and specifying group of important attributes) play an important role in model development. Today, data preprocessing is mainly based on human experiences and rules, which is expensive and might not work well in new domains. A possible alternative approach one can take is to design a classification model that can identify low quality data from the training set (e.g., through influence function-based approaches (Chen et al. 2021)).

Secondly, the AI generated descriptions are not fool proof. Thus, in order to ensure the users can have a reasonably good experience, post-processing of AI generated descriptions in the production platform is necessary to filter out any inconsistent or low quality contents. Currently, this step is often based on rules and involves humans in the loop. Nevertheless, it is a necessary consideration during deployment to uphold user experience.

Conclusions and Future Work

In this paper, we report our work on developing and deploying an automatic product copywriting generation system (APCG) for the JD e-commerce product recommendation platform. The proposed system consists of a transformer-pointer network and a pre-trained sequence-to-sequence model to perform natural language generation tasks in order to generate descriptions for products automatically. The system incorporates automatic evaluation and human screening modules to guard against the occasional low quality generated textual contents and ensure a high level of user experience. By adopting the APCG system, a high volume of 2.53 million product descriptions have been generated over a seven month period. The overall average CTR achieved by the JD Discover Goods Channel product recommendation platform is improved by 4.22%, while its CVR is improved by 3.61% on an year-on-year basis. The accumulated GMV made by our system is improved by 213.42%, compared to the start number in Feb 2021.

In future work, we will focus on enhancing the model performance and expanding the application scenarios:

- Domain-specific pre-trained models: our current experience shows that public pre-trained models can help enhance the quality of natural language generation. We will develop pre-trained models specific for the e-commerce domain to further improve model performance.
- Expending application scenarios: besides product description generation, automatic natural language generation can benefit other application scenarios as well. We will explore applying APCG to other application scenarios such as personalized product description generation and multi-product advertisement post generation.

Acknowledgments

Han Yu is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-RP-2020-019); the Nanyang Assistant Professorship (NAP); and the RIE 2020 Advanced Manufacturing and Engineering (AME) Programmatic Fund (No. A20G8b0102), Singapore. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2016. Neural Machine Translation by Jointly Learning to Align and Translate. arXiv:1409.0473.

Banko, M.; Mittal, V. O.; and Witbrock, M. J. 2000. Headline generation based on statistical translation. *ACL*, 318–325.

Bowman, S.; Angeli, G.; Potts, C.; and Manning, C. D. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*, 632–642.

Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Chen, Q.; Lin, J.; Zhang, Y.; Yang, H.; Zhou, J.; and Tang, J. 2019. Towards knowledge-based personalized product description generation in e-commerce. In *KDD*, 3040–3050.

Chen, Y.; Li, B.; Yu, H.; Wu, P.; and Miao, C. 2021. HyDRA: Hypergradient Data Relevance Analysis for Interpreting Deep Neural Networks. In *AAAI*, 7081–7089.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019a. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, 4171–4186.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019b. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *ACL*.

Dong, L.; Yang, N.; Wang, W.; Wei, F.; Liu, X.; Wang, Y.; Gao, J.; Zhou, M.; and Hon, H.-W. 2019. Unified Language Model Pre-training for Natural Language Understanding and Generation. arXiv:1905.03197.

Gehring, J.; Auli, M.; Grangier, D.; Yarats, D.; and Dauphin, Y. N. 2017. Convolutional Sequence to Sequence Learning. arXiv:1705.03122.

Keskar, N. S.; McCann, B.; Varshney, L. R.; Xiong, C.; and Socher, R. 2019. CTRL: A Conditional Transformer Language Model for Controllable Generation. arXiv:1909.05858.

Knight, K.; and Marcu, D. 2000. Statistics-based summarization-step one: Sentence compression. *AAAI*, 703–710.

Lafferty, J. D.; McCallum, A.; and Pereira, F. C. N. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML*, 282–289. San Francisco, CA, USA.

Lavie, A.; Sagae, K.; and Jayaraman, S. 2004. The significance of recall in automatic metrics for MT evaluation. In *AMTA*, 134–143.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Li, H.; Yuan, P.; Xu, S.; Wu, Y.; He, X.; and Zhou, B. 2020. Aspect-aware multimodal summarization for chinese e-commerce products. In *AAAI*, volume 34, 8188–8195.

Li, J.; Song, Y.; Zhang, H.; Chen, D.; Shi, S.; Zhao, D.; and Yan, R. 2018. Generating Classical Chinese Poems via Conditional Variational Autoencoder and Adversarial Training. In *EMNLP*, 3890–3900.

Lin, C.-Y. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, 74–81.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. Roberta: A robustly optimized bert pretraining approach. In *ACL*.

Och, F. J.; Gildea, D.; Khudanpur, S.; Sarkar, A.; Yamada, K.; Fraser, A.; Kumar, S.; Shen, L.; Smith, D.; Eng, K.; Jain, V.; Jin, Z.; and Radev, D. 2004. A Smorgasbord of Features for Statistical Machine Translation. In *NAACL-HLT*, 161–168.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 311–318.

Post, M. 2018. A call for clarity in reporting BLEU scores. *arXiv preprint arXiv:1804.08771*.

Radford, A.; Narasimhan, K.; Salimans, T.; and Sutskever, I. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language models are unsupervised multitask learners. *OpenAI Blog*, 1(8).

Rajpurkar, P.; Zhang, J.; Lopyrev, K.; and Liang, P. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *EMNLP*, 2383–2392.

Reiter, E.; and Dale, R. 2000. *Building Natural Language Generation Systems*. Studies in Natural Language Processing. Cambridge University Press.

Rush, A. M.; Chopra, S.; and Weston, J. 2015. A neural attention model for abstractive sentence summarization. *EMNLP*, 379–389.

Song, K.; Tan, X.; Qin, T.; Lu, J.; and Liu, T.-Y. 2019. MASS: Masked Sequence to Sequence Pre-training for Language Generation. In *ICML*, 5926–5936. PMLR.

Su, J. 2021. T5 PEGASUS - ZhuiyiAI. Technical Report 8.

Sutskever, I.; Vinyals, O.; and Le, Q. V. 2014. Sequence to Sequence Learning with Neural Networks. arXiv:1409.3215.

Tao, C.; Gao, S.; Shang, M.; Wu, W.; Zhao, D.; and Yan, R. 2018. Get The Point of My Utterance! Learning Towards Effective Responses with Multi-Head Attention Mechanism. In *IJCAI*, 4418–4424.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. arXiv:1706.03762.

Wang, J.; Hou, Y.; Liu, J.; Cao, Y.; and Lin, C.-Y. 2017. A statistical framework for product description generation. In *IJCNLP*, 187–192.

Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; and Philip, S. Y. 2020. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1): 4–24.

Yao, L.; Peng, N.; Weischedel, R.; Knight, K.; Zhao, D.; and Yan, R. 2019. Plan-and-Write: Towards Better Automatic Storytelling. *AAAI*, 33(01): 7378–7385.

Zhan, H.; Zhang, H.; Chen, H.; Shen, L.; Ding, Z.; Bao, Y.; Yan, W.; and Lan, Y. 2021. Probing Product Description Generation via Posterior Distillation. In *AAAI*, volume 35, 14301–14309.

Zhang, X.; Jiang, Y.; Shang, Y.; Cheng, Z.; Zhang, C.; Fan, X.; Xiao, Y.; and Long, B. 2021. DSGPT: Domain-Specific Generative Pre-Training of Transformers for Text Generation in E-commerce Title and Review Summarization. In *SIGIR*, 2146–2150.

Zhang, X.; Zhao, J.; and LeCun, Y. 2015. Character-level convolutional networks for text classification. *NeurIPS*, 28: 649–657.

Zhang, Y.; Sun, S.; Galley, M.; Chen, Y.-C.; Brockett, C.; Gao, X.; Gao, J.; Liu, J.; and Dolan, W. B. 2020. DIALOGPT: Large-Scale Generative Pre-training for Conversational Response Generation. In *ACL*, 270–278.

Zou, Y.; Zhang, X.; Lu, W.; Wei, F.; and Zhou, M. 2020. Pre-training for Abstractive Document Summarization by Reinstating Source Text. In *EMNLP*, 3646–3660.