Received: 26 January 2023 | Accepted: 21 February 2023

DOI: 10.1002/aaai.12084

SPECIAL TOPIC ARTICLE

Automatic product copywriting for e-commerce

Yanyan Zou[1] | Xueying Zhang[1] | Jing Zhou[1] | Shiliang Diao[1] | Jiajia Chen[1] | Zhuoye Ding[1] | Zhen He[1] | Xueqi He[1] | Yun Xiao[2] | Bo Long[1] | Mian Ma[1] | Sulong Xu[1] | Han Yu[3] | Lingfei Wu[2]

[1]JD.COM, Beijing, China

[2]JD.COM Silicon Valley Research Center, Mountain View, California, USA

[3]Nanyang Technological University, Singapore, Singapore

Correspondence
Yanyan Zou, JD.COM, Beijing, China.
Email: zoe.yyzou@gmail.com

Funding information
Nanyang Assistant Professorship (NAP); RIE 2020 Advanced Manufacturing and Engineering (AME) Programmatic Fund, Grant/Award Number: A20G8b0102; National Research Foundation Singapore, Grant/Award Number: AISG2-RP-2020-019

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 pretrained 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 February 2021. By September 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 February 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, thus in turn, promote products as well as increase sales. An example of product copywriting for "iPhone 13 Pro Max" released on the JD[1] e-commerce product recommendation platform is illustrated in Figure 1. Traditionally, product copywriting is performed by human copywriters, like professional writers and advertisers. However, such an approach has important limitations. First of all, the efficiency of human copywriters cannot match the growth rate of new products in e-commerce platforms, where millions of products are emerging each day. Second, manpower cost increases as the number of products in the system increases. Last but not least, specialist training and

FIGURE 1 An example of product copywriting on the JD e-commerce product recommendation platform. The original product copywriting is in Chinese, which is then translated in English for readability as highlighted in blue box.

tutoring are required for copywriters for different products and scenarios. To overcome these issues, automatic product copywriting generation (APCG) 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 nonlinguistic 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 highlight certain information; dialog generation Tao et al. (2018); Zhang et al. (2020b) 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 NLG 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 pretraining plus fine-tuning paradigm has gained traction and been widely applied in real-world applications. Under this paradigm, models are first pretrained on large-scale corpus Devlin et al. (2019a); Brown et al. (2020), or on domain-specific corpus Zhang et al. (2021); Zou et al. (2020) with well-defined pretraining tasks. The resulting pretrained models are then fine-tuned to adapt to different downstream tasks. Research in this paradigm mainly focuses on defining training objectives in the pretraining and fine-tuning stages. 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 pretraining and fine-tuning techniques have achieved outstanding performance in language generation tasks.

In this paper, we report on our experience deploying the proposed APCG system into the JD.com e-commerce product recommendation platform. It is made up of two main components: (1) a NLG module, which is built from a transformer–pointer network and a pretrained 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 product descriptions[2]. For various products from different domains (e.g., clothes and mobile phones), 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 to serve host promote products with generated advertising copywriting.

Since its deployment in JD.com in February 2021, APCG has generated 2.53 million product descriptions by September 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 attributes A, and the corresponding attribute values V for each attribute $a \in A$, as well as a short advertisement slogan s, which is usually a list of short phrases written by advertising experts (e.g., marketers and shoppers). 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 fabric type. 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*, the intended user feature can be *ladies*, and the fabric type can be *98% cotton, 2% lycra*. 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. Recall that an example of product copywriting is demonstrated in Figure 1.

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 bounded by the training corpus since the training samples are usually limited. Chen et al. 2019 designed a transformer-based text generation model to incorporate the external knowledge (i.e., Wikipedia knowledge base) into text generation process. Besides, the textual information (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 expensive 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 are 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 where the data points are very sparse. This could also reduce the quality of the generated product descriptions. On the other hand, each product is accompanied with several short advertising phrases written by experts.

Nevertheless, 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 as well as a pretrained sequence-to-sequence model for APCG in e-commerce settings. Both of such two models constitute our APCG system.

As illustrated in Figure 2, 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, and so forth. As shown in Figure 3, the overall system consists of the following steps:

(1) NLG model training: To collect data, including both product information (i.e., product titles, attributes, as well as advertising slogans) and descriptions, for training the NLG model. Different models are selected upon different scenarios. Both the transformer–pointer network and the pretrained 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.

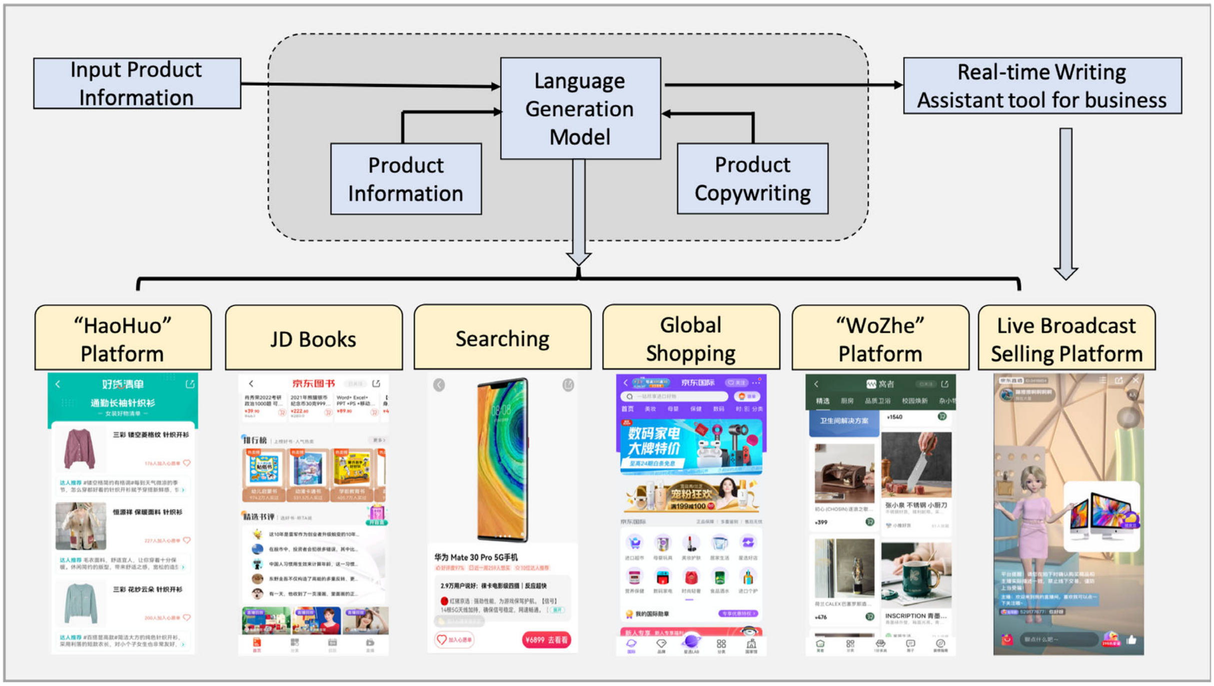

FIGURE 2 Overview of the proposed Automatic Product Copywriting Generation (APCG) system.

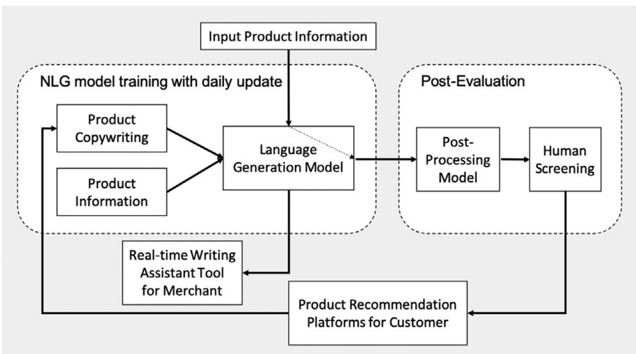

FIGURE 3 Workflow of the APCG system. APCG, Automatic Product Copywriting Generation.

USE OF AI TECHNOLOGY

In this section, we describe the key techniques adopted by the APCG AI Engine, including data collection and cleaning, the transformer–pointer network and pretrained language model for copywriting generation, as well as the post-processing modules.

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. As illustrated in Figure 1, the title is "iPhone 13 Pro Max," and the product copywriting is listed in blue box. In this work, our aim is to automatically generate the copywriting part, while the description title is not the focus. For simplicity, we use the "*copywriting/description title*" to denote such a title, and the "copywriting/description" to refer to the product advertising copywriting. 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 (if accessible). For each product 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. First, texts are extracted from the product detail images. Then, a pretrained language is used as a classification model to identify if a text from product detail image is appropriate as the input. To be specific, the extracted texts are ranked in descending order of their importance and relevance between itself and the product title as well as attributes 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, where an abstractive summarization model takes as input product title, attributes, OCR, as well as customer reviews, and generate a short description that highlight product's key characteristics with advertising words. We designed a pointer-generator network, which allows the transformer network to generate new words and copy words from the input text at 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 4, 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 multihead attention, we obtain the copy distributions by summing across multiple heads.

The pretrained 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, pretrained 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

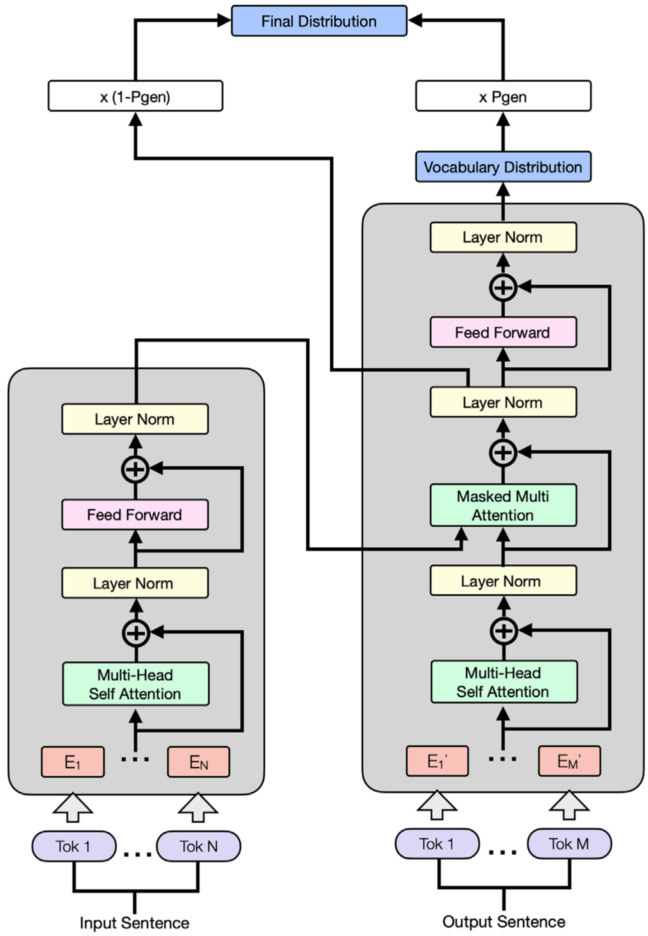

FIGURE 4 The overview of a transformer–pointer network structure.

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 NLG tasks (e.g., machine translation Song et al. (2019) and text summarization Zou et al. (2020)). This demonstrates the effectiveness and transferability of models pretrained on large scale unlabeled data. Recent research progresses in Brown et al. (2020) have demonstrated that pretrained models can play as an important role in few-shot learning with limited training instances.

In the product copywriting for e-commerce scenario, there are various categories of products (e.g., clothing, mobile phones, and sportswear). The product characteristics and data properties vary significantly across different categories. One straightforward solution is to train the transformer–pointer model on the specific training data for each category individually. However, this will incur high computation and maintenance costs since at least one model is required to be maintained for each category while we have hundreds of categories of products on the e-commerce platform. 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. Thanks to the development of pretraining techniques Song et al. (2019); Dong et al. (2019); Lewis et al. (2019), one pretrained model could well fit various downstream tasks (i.e., in our scenario, different product categories) and does not require much training data for fine-tuning. Therefore, we adopt the pretrained model for product copywriting, which could reduce the model maintenance cost.

Dong et al. (2019) (UniLM) proposed a unified language model that is pretrained using masked, unidirectional, and sequence-to-sequence language modeling objectives. Such a model can be used for both NLU and generation tasks. Song et al. (2019) proposed to pretrain a sequence-to-sequence transformer network by masking a span of text and then predicting the masked tokens. Zhang et al. (2020a) (PEGASUS) proposed to remove or mask sentences from an input document and learn to generate such removed or masked sentences for model pretraining. Lewis et al. (2019) proposed to combine the two tasks of text infilling and sentence permutation as a single objective for sequence-to-sequence transformer pretraining. Raffel et al. (2020) (T5) studies different pretraining objectives, model architectures, and unlabeled datasets. ProphetNet Qi et al. (2020) pretrained a model by predicting the next n tokens simultaneously.

The above recent studies have shown their advantages of various NLU and generation tasks. However, such models are developed specific for English and general domains, while in our scenarios, we would like to develop a pretrained language model for Chinese e-commerce domains. To achieve such a goal, 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 $X = (x_1, x_2, \dots, x_{|X|})$ paired with its corresponding output sequence $Y = (y_1, y_2, \dots, y_{|Y|})$, we aim to learn the model parameters θ and estimate the conditional probability:

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

where $y_{<t}$ stands for all tokens before position t (i.e., $y_{<t} = (y_1, y_2, \dots, y_{t-1})$). 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:

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

We pretrain 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 pretraining.

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 pretrained 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 pretrained, we fine tune the model on the parallel copywriting training data, consisting of instances from all categories.

It is worth noting that the data for model pretraining is collected from JD.com e-commerce, where the human-written product copywriting is extracted. In other words, a piece of complete product copywriting is a document here.

During pretraining, we consider two settings. **Setting one**: pretraining a model with one single objective, that is, either sentence re-ordering or pseudo summary generation, resulting in two different pretrained models. **Setting two**: employing both objectives. For each training batch, we randomly choose one objective and each objective is used for 1/2 of the training time, obtaining one model. Based on our preliminary results, the setting two results in a model with better performance, so we deploy our model with the latter setting.

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 (in short, TP) and pretraining-based copywriting generation model (in short, PCG) with the following two baselines before making deployment decisions: the template-based generation model (i.e., Template) Wang et al. (2017); 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. The ROUGE scores are computed using the `ROUGE-1.5.5.pl` script[3]. We report the SacreBLUE, ROUGE-1, ROUGE-2, ROUGE-L, BLEU-1, BLEU-2, BLUE-3, BLEU-4, and Meteor scores, as listed in Table 1. 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 automatic evaluation metrics.

Since descriptions generated by models may produce incoherent or grammatically incorrect outputs, we also evaluated these four models by eliciting human judgements. We invited 10 human participants who are native speaker in Chinese. The volunteers 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 the following criteria: Fluency: Is the description grammatically sound? Relevance: Is the description relevant to the product? Diversity: Does the text describe the product from different perspectives?

The results are listed in Table 2. 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 APCG 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%.

TABLE 1 Offline evaluation results for the four models.

Model	SacreBLEU	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*

*denotes a value is significantly different from the second highest one with $p < 0.05$.

TABLE 2 Human 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**

*denotes a value is significantly different from the second highest one with $p < 0.05$.

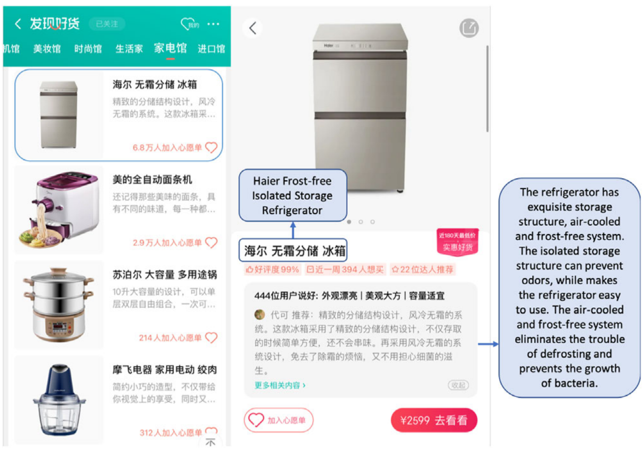

FIGURE 5 Examples of APCG generated descriptions on the JD e-commerce product recommendation platform. APCG, Automatic Product Copywriting Generation.

The usage of the JD product recommendation platform generates new data daily, our NLG models are trained and updated daily with these new data.

As illustrated in Figure 3, 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 5 shows an example of the product recommendation platform for customers, "HaoHuo." 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 6 shows the use case in the live broadcast selling platform. By September 2021, the APCG writing assistant tool has been adopted by more than 2000 merchants with daily live streams up to 150 sessions.

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 3 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.

FIGURE 6 Examples of APCG generated descriptions on the JD live broadcast selling platform. The virtual salesperson reads the product descriptions in the virtual studio. APCG, Automatic Product Copywriting Generation.

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 February 2021.

TABLE 3 Latency test comparison of CPU and GPU services.

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

APPLICATION USE AND PAYOFF

The APCG system has been deployed in JD.com since February 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 CTR and CVR. CTR is defined as

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

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

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

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 Pretrained 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 pretrained 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 pretrained model incurs high computation costs. Thus, during APCG deployment, PCG mainly focuses on categories with limited training data or newly released products.

Between February 2021 and September 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 GMV made by our system is improved by 213.42%, compared to the start number in February 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 pretrained 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 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.

First, 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)).

Second, the AI-generated descriptions are not fool proof. Thus, in order to ensure that 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 APCG system for the JD e-commerce product recommendation platform. The proposed system consists of a transformer–pointer network and a pretrained sequence-to-sequence model to perform NLG 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 has been generated over a 7-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 February 2021.

In future work, we will focus on enhancing the model performance and expanding the application scenarios: Domain-specific pretrained models: our current experience shows that public pretrained models can help enhance the quality of NLG. We will develop pretrained models specific for the e-commerce domain to further improve model performance. Personalized copywriting: the deployed generation models do not consider the personalized preferences of various customers.

In practice, users might focus on different aspects of a product. Thus, the personalized copywriting is necessary to further improve user experience. Expending application scenarios: besides product description generation, automatic NLG can benefit other application scenarios as well. We will explore applying APCG to other application scenarios such as personalized product description generation and multiproduct 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.

CONFLICT OF INTEREST STATEMENT

The authors declare that there is no conflict.

ORCID

Yanyan Zou https://orcid.org/0000-0002-8153-9299
Han Yu https://orcid.org/0000-0001-6893-8650

ENDNOTES

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

[2] For simplicity, we interchangeably use product copywriting and description in this paper.

[3] https://github.com/bheinzerling/pyrouge.git

REFERENCES

Bahdanau, D., K. Cho, and Y. Bengio. 2016. "Neural Machine Translation by Jointly Learning to Align and Translate." Technical report.

Banko, M., V. O. Mittal, and M. J. Witbrock. 2000. "Headline Generation Based on Statistical Translation." In *ACL*, 318–25.

Bowman, S., G. Angeli, C. Potts, and C. D. Manning. 2015. "A Large Annotated Corpus for Learning Natural Language Inference." In *EMNLP*, 632–42.

Brown, T. B., B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, et al. 2020. "Language Models are Few-shot Learners." In NeurIPS.

Chen, Q., J. Lin, Y. Zhang, H. Yang, J. Zhou, and J. Tang. 2019. "Towards Knowledge-Based Personalized Product Description Generation In E-Commerce." In *KDD*, 3040–50.

Chen, Y., B. Li, H. Yu, P. Wu, and C. Miao. 2021. "HyDRA: Hypergradient Data Relevance Analysis for Interpreting Deep Neural Networks." In *AAAI*, 7081–9.

Devlin, J., M.-W. Chang, K. Lee, and K. Toutanova. 2019a. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In *NAACL*, 4171–86.

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

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

Gehring, J., M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin. 2017. "Convolutional Sequence to Sequence Learning." In ICML.

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

Knight, K., and D. Marcu. 2000. "Statistics-Based Summarization-Step One: Sentence Compression." In *AAAI*, 703–10.

Lafferty, J. D., A. McCallum, and F. C. N. Pereira. 2001. "Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data." In *ICML*, 282–9 San Francisco, CA, USA.

Lavie, A., K. Sagae, and S. Jayaraman. 2004. "The Significance of Recall in Automatic Metrics for MT Evaluation." In *AMTA*, 134–43.

Lewis, M., Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. 2019. "Bart: Denoising Sequence-to-sequence Pre-training for Natural Language Generation, Translation, and Comprehension." arXiv preprint arXiv:1910.13461.

Li, H., P. Yuan, S. Xu, Y. Wu, X. He, and B. Zhou. 2020. "Aspect-Aware Multimodal Summarization for Chinese E-Commerce Products." In *AAAI*, volume 34, 8188–95.

Li, J., Y. Song, H. Zhang, D. Chen, S. Shi, D. Zhao, and R. Yan. 2018. "Generating Classical Chinese Poems via Conditional Variational Autoencoder and Adversarial Training." In *EMNLP*, 3890–900.

Lin, C.-Y. 2004. "Rouge: A package for Automatic Evaluation of Summaries." In *Text Summarization Branches Out*, 74–81.

Liu, Y., M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. 2019. "Roberta: A Robustly Optimized Bert Pretraining Approach." In *ACL*.

Och, F. J., D. Gildea, S. Khudanpur, A. Sarkar, K. Yamada, A. Fraser, S. Kumar, L. Shen, D. Smith, K. Eng, V. Jain, Z. Jin, and D. Radev. 2004. "A Smorgasbord of Features for Statistical Machine Translation." In *NAACL-HLT*, 161–8.

Papineni, K., S. Roukos, T. Ward, and W.-J. Zhu. 2002. "Bleu: A Method for Automatic Evaluation of Machine Translation." In *ACL*, 311–8.

Post, M. 2018. "A Call for Clarity in Reporting BLEU Scores." In WMT.

Qi, W., Y. Yan, Y. Gong, D. Liu, N. Duan, J. Chen, R. Zhang, and M. Zhou. 2020. "ProphetNet: Predicting Future N-gram for Sequence-to-Sequence Pre-training." In *EMNLP*, 2401–10.

Radford, A., K. Narasimhan, T. Salimans, and I. Sutskever. 2018. "Improving Language Understanding by Generative Pre-training." https://s3-us-west-2.amazonaws.com/openai-assets/researchcovers/languageunsupervised/languageunderstandingpaper.pdf.

Radford, A., J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. 2019. "Language Models are Unsupervised Multitask Learners." *OpenAI Blog* 1(8).

Raffel, C., N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. 2020. "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer." *Journal of Machine Learning Research* 21(140): 1–67.

Rajpurkar, P., J. Zhang, K. Lopyrev, and P. Liang. 2016. “SQuAD: 100,000+ Questions for Machine Comprehension of Text.” In *EMNLP*, 2383–92.

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

Rush, A. M., S. Chopra, and J. Weston. 2015. “A Neural Attention Model for Abstractive Sentence Summarization.” In *EMNLP*, 379–89.

Song, K., X. Tan, T. Qin, J. Lu, and T.-Y. Liu. 2019. “MASS: Masked Sequence to Sequence Pre-training for Language Generation.” In *ICML*, 5926–36, PMLR.

Su, J. 2021. “T5 PEGASUS—ZhuiyiAI.” Technical report 8.

Sutskever, I., O. Vinyals, and Q. V. Le. 2014. “Sequence to Sequence Learning with Neural Networks.” In NeurIPS.

Tao, C., S. Gao, M. Shang, W. Wu, D. Zhao, and R. Yan. 2018. “Get The Point of My Utterance! Learning Towards Effective Responses with Multi-Head Attention Mechanism.” In *IJCAI*, 4418–24.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. 2017. “Attention Is All You Need.” In NeurIPS.

Wang, J., Y. Hou, J. Liu, Y. Cao, and C.-Y. Lin. 2017. “A Statistical Framework for Product Description Generation.” In *IJCNLP*, 187–92.

Wu, Z., S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip. 2020. “A Comprehensive Survey on Graph Neural Networks.” *IEEE Transactions on Neural Networks and Learning Systems* 32(1): 4–24.

Yao, L., N. Peng, R. Weischedel, K. Knight, D. Zhao, and R. Yan. 2019. “Plan-and-Write: Towards Better Automatic Storytelling.” In *AAAI* 33(01): 7378–85.

Zhan, H., H. Zhang, H. Chen, L. Shen, Z. Ding, Y. Bao, W. Yan, and Y. Lan. 2021. “Probing Product Description Generation via Posterior Distillation.” In *AAAI*, volume 35, 14301–9.

Zhang, J., Y. Zhao, M. Saleh, and P. Liu. 2020a. “Pegasus: Pre-training with Extracted Gap-Sentences for Abstractive Summarization.” In *ICML*, 11328–39, PMLR.

Zhang, X., Y. Jiang, Y. Shang, Z. Cheng, C. Zhang, X. Fan, Y. Xiao, and B. Long. 2021. “DSGPT: Domain-Specific Generative Pre-Training of Transformers for Text Generation in E-commerce Title and Review Summarization.” In *SIGIR*, 2146–50.

Zhang, X., J. Zhao, and Y. LeCun. 2015. “Character-Level Convolutional Networks for Text Classification.” In *NeurIPS* 28: 649–57.

Zhang, Y., S. Sun, M. Galley, Y.-C. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, and W. B. Dolan. 2020b. “DIALOGPT: Large-Scale Generative Pre-training for Conversational Response Generation.” In *ACL*, 270–8.

Zou, Y., X. Zhang, W. Lu, F. Wei, and M. Zhou. 2020. “Pre-Training for Abstractive Document Summarization by Reinstating Source Text.” In *EMNLP*, 3646–60.

How to cite this article: Zou, Y., X. Zhang, J. Zhou, S. Diao, J. Chen, Z. Ding, Z. He, X. He, Y. Xiao, B. Long, M. Ma, S. Xu, H. Yu, and L. Wu. 2023. “Automatic product copywriting for e-commerce.” *AI Magazine* 44: 41–53. https://doi.org/10.1002/aaai.12084

AUTHOR BIOGRAPHIES

Yanyan Zou is an Applied Scientist in Recommendation Platform at JD.com, launching cutting-edge AI models into practical productions. Her research interests primarily lie in the areas of natural language processing, especially the natural language generation and pretraining models, with several papers published in top-tier conferences (e.g., ACL, EMNLP, and NAACL). She received her B.Engr. degree in 2015 from Xiamen University, China, as well as her Ph.D. degree from Singapore University of Technology and Design (SUTD) in 2020. She has served as a PC Member/Reviewer of the following major AI/ML/NLP conferences, including ACL, EMNLP, NAACL, AAAI, IJCAI, and so forth.

Xueying Zhang is a Research Scientist in TikTok, with 5+ years of industry experience in AIGC, building and deploying machine/deep learning models to production with publications on top conferences. Previously, she was an Applied Scientist at JD.com Silicon Valley Research Center. Her research interest focuses in deep learning on NLG, NLP, and multimodal problems. Before that, she was a Research Scientist at an AI startup company at Silicon Valley and led a 5+ software engineer team for developing NLG projects for real-world applications. She received her B.S. degree in 2011 from Harbin Institute of Technology, China, and her Ph.D degree from The University of Tennessee in 2017. She has published over 15 peer-reviewed papers in top conferences and journals.

Jing Zhou is a Machine Learning Architecture Engineer at Search and Recommendation Platform in JD.com. She received her Master degree from Harbin Institute of Technology, China in 2015.

Shiliang Diao is a Product Manager at JD.com, working to improve user experience at JD e-commerce platform.

Jiajia Chen is a Product Manager at JD.com, working to improve user experience at JD e-commerce platform.

Zhuoye Ding leads the Recommendation Platform in JD.com, supporting all recommendation algorithms and strategies at JD e-commerce platform.

Zhen He leads the Recommendation Product Team at JD.com, supporting product strategies for JD.com e-commerce platform.

Xueqi He is an Engineer Manager at Search and Recommendation Platform in JD.com.

Yun Xiao is a Head of Search, JD.COM Silicon Valley Research Center. Before joining in JD, he was a Senior Engineer Manager in Yahoo!.

Bo Long is an Engineer Leader in the Recommendation and Foundation AI for Monetization at Meta, where they support an amazing team of Researchers/Engineers to build AI technology in various areas, including Graph Learning, User Understanding, Large Scale Training, AutoML, Reinforcement Learning, Semi & Self-Supervised Learning, and so forth, for various monetization applications from Facebook, Instagram, WhatsApp, Reels, and Marketplace products. Previously, he was the Vice President of JD Group, Head of Search and Recommendation Business Unit, supporting an amazing talents of product teams, infra teams, data teams, ML/AI teams to create the best user experience at JD e-commerce platforms.

Mian Ma is an Engineer Manager at Recommendation Platform in JD.com, supporting all recommendation algorithms and strategies at JD e-commerce platform.

Sulong Xu is the Head of Search and Recommendation Platform in JD.com, which designs search and recommendation algorithms and strategies for JD e-commerce platforms.

Han Yu is a Nanyang Assistant Professor (NAP) in the School of Computer Science and Engineering (SCSE), Nanyang Technological University (NTU), Singapore. He held the prestigious Lee Kuan Yew Post-Doctoral Fellowship (LKY PDF) from 2015 to 2018. He obtained his PhD from the School of Computer Science and Engineering, NTU. His research focuses on federated learning and algorithmic fairness. He has published over 200 research papers and book chapters in leading international conferences and journals. He is a co-author of the book Federated Learning—the first monograph on the topic of federated learning. His research works have won multiple awards from conferences and journals. He is a Senior Member of AAAI, CCF, and IEEE.

Lingfei Wu is an Engineering Manager in the Content and Knowledge Graph Group at Pinterest, where they are building the next generation Knowledge Graph to empower Pinterest recommendation/research systems across all major surfaces including Homefeed, Search, Ads, and so forth. He earned his Ph.D. degree in Computer Science from the College of William and Mary in 2016. Previously, he was a Principal Scientist at JD.COM Silicon Valley Research Center, leading a team of 30+ Machine Learning/Natural Language Processing Scientists and Software Engineers to build intelligent e-commerce personalization systems. Before that, he was a research staff member at IBM Thomas J. Watson Research Center and led a 10+ Research Scientist team for developing novel Graph Neural Networks methods and systems, which leads to the #1 AI Challenge Project in IBM Research and multiple IBM Awards including three-time Outstanding Technical Achievement Award. He has published one book (in GNNs) and more than 100 top-ranked conference and journal papers, and is a co-inventor of more than 40 filed US patents. Because of the high commercial value of his patents, he received eight invention achievement awards and was appointed as IBM Master Inventors, class of 2020. He was the recipients of the Best Paper Award and Best Student Paper Award of several conferences such as IEEE ICC'19, AAAI workshop on DLGMA'20, and KDD workshop on DLG'19. His research has been featured in numerous media outlets, including NatureNews, YahooNews, AP News, PR Newswire, The Time Weekly, Venturebeat, MIT News, IBM Research News, and SIAM News. He has served as Industry and Government Program Co-Chairs of IEEE BigData'22, Sponsorship Co-Chairs of KDD'22, and Associate Conference Co-Chairs of AAAI'21 and is the Founding Co-Chairs for several workshops such as Deep Learning on Graphs (with AAAI'20-22 and KDD'19-22). He has also served as Associate Editor for IEEE Transactions on Neural Networks and Learning Systems and ACM Transactions on Knowledge Discovery from Data.