MixPAVE: Mix-Prompt Tuning for Few-shot Product Attribute Value Extraction

Li Yang[1]*, Qifan Wang[2]*, Jingang Wang[3]†, Xiaojun Quan[4], Fuli Feng[5], Yu Chen[2], Madian Khabsa[2], Sinong Wang[2], Zenglin Xu[6] and Dongfang Liu[7]†

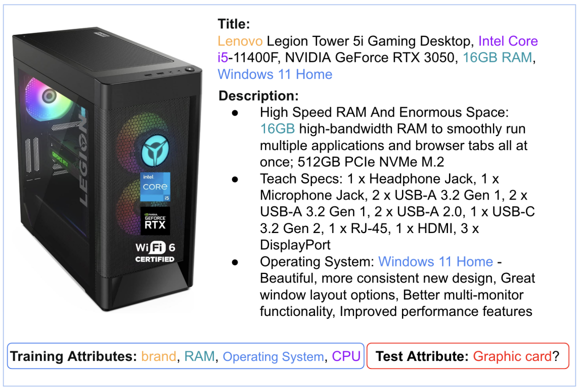
[1]Google Research [2]Meta AI [3]Meituan Lab [4]Sun Yat-sen University
[5]University of Science and Technology of China [6]Peng Cheng Lab
[7]Rochester Institute of Technology
lyliyang@google.com wqfcr@fb.com

Abstract

The task of product attribute value extraction is to identify values of an attribute from product information. Product attributes are important features, which help improve online shopping experience of customers, such as product search, recommendation and comparison. Most existing works only focus on extracting values for a set of known attributes with sufficient training data. However, with the emerging nature of e-commerce, new products with their unique set of new attributes are constantly generated from different retailers and merchants. Collecting a large number of annotations for every new attribute is costly and time consuming. Therefore, it is an important research problem for product attribute value extraction with limited data. In this work, we propose a novel prompt tuning approach with **Mix**ed **P**rompts for few-shot **A**ttribute **V**alue **E**xtraction, namely MixPAVE. Specifically, MixPAVE introduces only a small amount ($< 1\%$) of trainable parameters, i.e., a mixture of two learnable prompts, while keeping the existing extraction model frozen. In this way, MixPAVE not only benefits from parameter-efficient training, but also avoids model overfitting on limited training examples. Experimental results on two product benchmarks demonstrate the superior performance of the proposed approach over several state-of-the-art baselines. A comprehensive set of ablation studies validate the effectiveness of the prompt design, as well as the efficiency of our approach.

1 Introduction

Product attributes are important features associated with products, which form an essential component of e-commerce platforms (Nguyen et al., 2020; Yu et al., 2021). These attributes contain detailed information of the products, and provide useful guidance for customers to compare products and make purchasing decisions. They also facilitate merchants on various applications, including product recommendations (Truong et al., 2022), product search (Lu et al., 2021), and product question answering (Zhang et al., 2020b; Rozen et al., 2021). Therefore, product attribute value extraction has recently attracted a lot of interest from both academia and industry, with a plethora of research (Putthividhya and Hu, 2011a; Chen et al., 2019; Zhang et al., 2022; Wang et al., 2022; Shinzato et al., 2022) being developed to solve this problem.



Title:
Lenovo Legion Tower 5i Gaming Desktop, Intel Core i5-11400F, NVIDIA GeForce RTX 3050, 16GB RAM, Windows 11 Home

Description:

- High Speed RAM And Enormous Space: 16GB high-bandwidth RAM to smoothly run multiple applications and browser tabs all at once; 512GB PCIe NVMe M.2
- Teach Specs: 1 x Headphone Jack, 1 x Microphone Jack, 2 x USB-A 3.2 Gen 1, 2 x USB-A 3.2 Gen 1, 2 x USB-A 2.0, 1 x USB-C 3.2 Gen 2, 1 x RJ-45, 1 x HDMI, 3 x DisplayPort
- Operating System: Windows 11 Home - Beautiful, more consistent new design, Great window layout options, Better multi-monitor functionality, Improved performance features

Training Attributes: brand, RAM, Operating System, CPU

Test Attribute: Graphic card?

Figure 1: An example of a “desktop” product from MAVE dataset. There are multiple attributes with their corresponding values in the product during training. We want the model to adapt quickly to new attributes.

Existing works (Zheng et al., 2018; Xu et al., 2019; Yan et al., 2021; Yang et al., 2022) mostly learn extraction models to extract values for a predefined set of attributes, assuming all attributes are covered in the training examples. However, in real-world scenarios, new products are emerging everyday with their unique set of new attributes. In fact, even within the existing products, there are new attributes being generated by the merchants. For example, Figure 1 shows a “desktop” product, where the values corresponding to the attributes of “brand”, “RAM”, “Operating System” and “CPU” are annotated in the product title and description,

*Equal Contribution
†Corresponding Authors

Findings of the Association for Computational Linguistics: ACL 2023, pages 9978–9991
July 9-14, 2023 ©2023 Association for Computational Linguistics

but the "Graphic Card" attribute is not included in the training attributes. In this case, a desired model should quickly adapt to new attributes by providing a few examples.

One straightforward solution is to merge the new attributes with existing ones, and retrain the extraction model. The main drawback of this approach is that the full model needs to be retrained on all training data every time when there are new attributes, making it computational expensive and thus impractical to use. An alternative approach is to conduct full fine-tuning of the existing model on the small data associated with the new attributes. This approach is more data-efficient compared to the retraining with all data. However, the fine-tuned model is likely to overfit to the new attributes (due to very few training examples) especially for large-scale Transformer models (Wang et al., 2020; Yang et al., 2022). Another popular approach is to fine-tune only a subset of the parameters, such as the extraction head (Yosinski et al., 2014) or the bias terms (Cai et al., 2020). Prior research has also attempted at adding extra blocks or adapters (Pfeiffer et al., 2020) to the existing model. However, in general these strategies under-perform the full fine-tuned model.

Inspired by the recent advances on prompt tuning (Lester et al., 2021; Jia et al., 2022; He et al., 2022c; Ma et al., 2022), in this paper, we propose a novel prompt tuning approach with **Mix**ed **P**rompts for few-shot **A**ttribute **V**alue **E**xtraction, namely MixPAVE. In contrast to previous fine-tuning or partial tuning approaches, we introduce two sets of learnable prompts, textual prompts and key-value prompts, and augment the Transformer-based extraction model with the mixture of these two prompts. Specifically, textual prompts are prepended to the input sequence of each Transformer layer while key-value prompts are inserted to the key and value matrices in the self-attention block. Then MixPAVE is then learned by only fine-tuning the prompts (less than 1% of all parameters) on the new attributes, while keeping the other parameters in the Transformer frozen. Therefore, our approach not only benefits from parameter-efficient training, but also avoids model overfitting with limited training examples. Evaluations on two product datasets show the superior performance of our model over several state-of-the-art methods. The experimental results also demonstrate the effectiveness and efficiency of the proposed prompts in the few-shot scenarios. We summarize the main contributions as follows:

- We explore a different route in this work by proposing a novel prompt tuning approach for product attribute value extraction. Our approach enables fast adaptation of the existing extraction model to new attributes with just a few annotations.

- We design two types of prompts in our model by adding textual prompts to the input sequence, and inserting key-value prompts to the self-attention computation. These two prompts effectively and efficiently guide the model fine-tuning in the few-shot setting.

- We conduct comprehensive experiments on two product benchmarks, demonstrating the effectiveness of the proposed approach over several state-of-the-art partial model tuning and prompt tuning methods.

2 Related Work

2.1 Attribute Value Extraction

Early works in attribute value extraction include rule-based extraction methods (Vandic et al., 2012; Gopalakrishnan et al., 2012) and named entity recognition (NER) based approaches (Putthividhya and Hu, 2011b; Brooke et al., 2016), which suffer from limited coverage and closed world assumptions. With the advent of deep learning, various neural network methods (Huang et al., 2015; Zheng et al., 2018) are proposed, which formulate the problem as a sequential tagging problem. SUOpenTag (Xu et al., 2019) scales up these models by jointly encoding the attribute and product context. AdaTag (Yan et al., 2021) utilizes a mixture-of-experts (MoE) to parameterize its decoder with pre-trained attribute embeddings. AVEQA (Wang et al., 2020) and MAVEQA (Yang et al., 2022) leverage the recent Transformer (Vaswani et al., 2017) and BERT (Devlin et al., 2019) models by reformulating the problem as a question answering task. Meanwhile, TXtract (Karamanolakis et al., 2020) brings a taxonomy-aware approach to aid attribute extraction. OA-Mine (Zhang et al., 2022) proposes a framework that first generates attribute value candidates and then groups them into clusters of attributes in an open-world setting.

Several recent works (Singh et al., 2019; Tan and Bansal, 2019; Hu et al., 2020) explore the product visual features for enhancing the attribute value extraction. MJAVE (Zhu et al., 2020) introduces a multimodal model that predicts product attributes and extract values together from both product title and image. PAM (Lin et al., 2021) combines product descriptions, Optical Character Recognition (OCR) and visual information from the product, and fuses the three modalities into a multimodal Transformer. SMARTAVE (Wang et al., 2022) designs a structured multimodal Transformer to better encode the correlation among different product modalities. Despite achieving promising results, most of these methods learn to extract values for a predefined set of attributes, assuming all attributes are covered in the training examples. However, with the emerging nature of e-commerce, new products with their unique set of new attributes are constantly generated from different retailers and merchants.

2.2 Parameter Efficient Models

Parameter efficient methods (Guo et al., 2021; He et al., 2022a; Hu et al., 2022) are widely applied in training models with limited data. It is also commonly adopted in domain adaptation scenarios. Among these methods, partial tuning approaches (Yosinski et al., 2014; He et al., 2022b) fine-tune the last few layers of the backbone while freezing the others. Side-tuning (Zhang et al., 2020a) trains a "side" network and linear interpolates between pre-trained features and side-tuned features before being fed into the head. BitFit (Zaken et al., 2022) and TinyTL (Cai et al., 2020) fine-tune only the bias terms of a pre-trained backbone. Adapters (Houlsby et al., 2019; Pfeiffer et al., 2020, 2021) insert extra lightweight MLP modules with residual connection inside Transformer layers and only fine-tune those added modules.

Prompting (Liu et al., 2021) has been originally proposed for fast model adaptation in few-shot or zero-shot settings (Brown et al., 2020), which prepends text instruction to the input text on downstream tasks. Recent prompt tuning works (Lester et al., 2021; Li and Liang, 2021; He et al., 2022c; Ma et al., 2022) propose to treat the prompts as task-specific continuous vectors, and directly learn them during fine-tuning. Different from full fine-tuning, they achieve comparable performance but with much less parameter storage. However, most of these methods only simply add prompts to the input layer, which greatly limited their performances.

3 MixPAVE

3.1 Preliminary

Attribute Value Extraction Given product context and the attribute, the goal of attribute value extraction is to identify the value from the context. The product context is a text sequence describing the product, e.g., a concatenation of product title and description, denoted as $C = (c_1, c_2, \ldots, c_n)$. The attribute is denoted as $A = (a_1, \ldots, a_m)$, e.g., "Operating System" in Figure 1. The extraction model seeks the best text spans from the context that correspond to the attribute value.

There are various attribute extraction models tackling this problem. Several recent Transformer-based approaches (Wang et al., 2020; Yang et al., 2022) achieve state-of-the-art performance in extracting attribute values from product text. They formulate the problem as a question answering task by treating each attribute as a question, and identifying the answer (attribute value) from the product context. Specifically, as shown in Figure 2 (the frozen part), a Transformer encoder is used to jointly encode the attribute A and the product context C. Then an extraction head, e.g., sequential tagging, is applied to the output embeddings of the encoder to find the target spans. In this work, we use (Yang et al., 2022) as our backbone model with a T5 (Raffel et al., 2020) encoder.

Prompt Tuning Prompt tuning methods (Lester et al., 2021; Li and Liang, 2021) are proposed as a group of parameter efficient models for fast adaptation of large-scale pre-trained language models to downstream tasks. They introduce a set of task-specific prompts or prompt tokens, and prepend them to the input sequence. During fine-tuning, these prompts are learned on the data from the downstream task while freezing the backbone. Prompt tuning achieves promising results compared to other parameter efficient methods in few-shot settings.

3.2 Model Overview

In this section, we first define the problem of few-shot product attribute value extraction, and then provide an overview of the MixPAVE model. Assuming we have a backbone extraction model, $\mathbf{B}$, which is well-trained on a large set of attributes and values. Given a new attribute with a few annotations (usually less than 100 examples), the goal is to learn a model $\hat{\mathbf{B}}$ that can achieve good perfor-

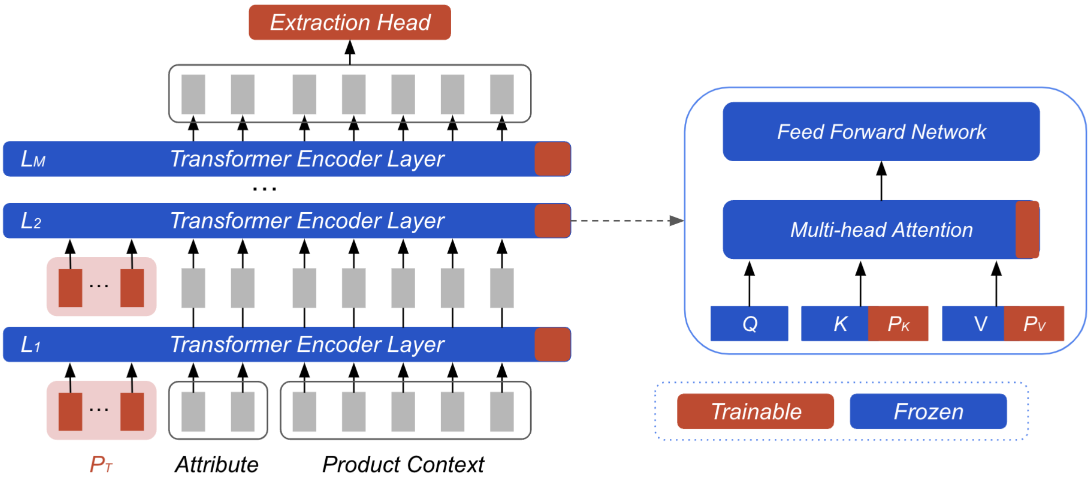


Figure 2: Overview of MixPAVE model on few-shot product attribute value extraction. We introduce two sets of trainable prompts during model fine-tuning. (1) Textual prompts P_T are prepended to the input sequence of each Transformer layer. (2) Key-value prompts P_K and P_V are added to the key and value matrices in the multi-head attention computation. It can be seen that most of the model parameters are frozen.

mance on the new attribute, while keeping same performance on existing attributes (i.e., not overfit to the new attribute). To solve this problem, we propose a new prompt tuning approach.

The overall model architecture of MixPAVE is shown in Figure 2. Essentially, our model only allows the extraction head (very light-weight) in the backbone to be trained during fine-tuning while freezing all other parameters. To maximize the model performance, we introduce two sets of trainable prompts and insert them into the backbone. (1) Textual prompts, P_T, are inserted to the input sequence of each encoder layer, which learns the extraction task for the new attribute. (2) Key-value prompts, P_K and P_V, are concatenated with the key and value parameter matrices in the attention module respectively, which learn the new attention pattern from the new data.

3.3 Textual Prompt

Textual prompts are a set of d-dimensional embedding vectors that have the same dimensionality with the input tokens. They are prepended to the input sequence of each Transformer encoder layer and interact with all the input tokens. Textual prompts play a similar role to those prompt tokens in traditional prompt tuning methods (Lester et al., 2021; Li and Liang, 2021), which learn task-specific embeddings to guide the model performing extraction task on the new attribute.

Formally, these textual prompts are defined as $P_T = \{P_T^1, P_T^2, \ldots, P_T^M\}$, where P_T^i denotes the learnable textual prompts in the i_{th} encoder layer, and M is the total number of layers. Then the encoder layers are represented as:

$$\begin{aligned} Z^1 &= L_1(P_T^1,\ A,\ C) \\ Z^i &= L_i(P_T^i,\ Z^{i-1}) \quad i = 2, 3, \ldots, M \end{aligned} \tag{1}$$

where Z^i represents the contextual embeddings of the attribute and product context computed by the i_{th} encoder layer. The different colors indicate trainable and frozen parameters, respectively. For the embeddings of the attribute and context tokens, A and C, they are initialized with token embeddings from the backbone.

3.4 Key-value Prompt

Textual prompts effectively learn the knowledge about the extraction task on the new attribute. However, they are not able to guide the information interaction within each encoder layer. When fine-tuning on new attributes with new data, the word distribution could be very different from those examples for training the backbone model. For instance, the fine-tuning data contains new values corresponding to the new attribute, with different sentence structures and presentations. Therefore, we need to increase the model capability to capture new information in the fine-tuning data, and conduct better attention among the input tokens for learning the new patterns.

To this end, we propose a novel set of key-value prompts, which are inserted to the attention block

Splits	AE-110K				MAVE				
	Lenses Color	Wheel Material	Product Type	REST	Device Type	Boot Style	Resolution	Compatibility	REST
Train	10	10	10	109K	100	100	100	100	2.6M
Test	165	118	368	-	20,273	16,459	14,437	6,728	-

Table 1: Statistics of the training and test examples within different few-shot attributes.

inside each encoder layer. Specifically, these key-value prompts, P_K and P_V, are small matrices (with a few columns) that have the same number of rows as the original key and value matrices in the attention block. They are then concatenated respectively to perform the new attention computations:

$$L(\cdot) = \text{FFN}\ (\text{MHA}\ (\cdot))$$
$$\text{MHA}(\cdot) = concat(softmax(\frac{Q_j K_j'^{T}}{\sqrt{d}})V_j') \tag{2}$$

where FFN is the feed-forward network and MHA is the multi-head attention inside the encoder layer. j represents the j_{th} head. K' and V' are the new key and value embedding matrices defined as:

$$\begin{aligned} K' &= concat(K,\ P_K) \\ V' &= concat(V,\ P_V) \end{aligned} \tag{3}$$

where K and V represent the original key and value matrices in the backbone. In this way, the key-value prompts can help guide the model adaptation to the new data.

3.5 Extraction Head

The extraction layer is essentially a sequential tagging (Xu et al., 2019) module, which extracts the final text span by assigning the $\{\text{Begin}, \text{Inside}, \text{Outside}, \text{End}\}$ tags to each tokens based on their embeddings obtained from the encoder, followed by a CRF layer (Yan et al., 2021):

$$T = \text{CRF}(softmax(W_T Z^M)) \tag{4}$$

where Z^M is the output embedding from the top layer of the encoder. W_T is the parameter matrix that projects the embeddings to the output logits, which is trainable in our model.

3.6 Discussion

MixPAVE is a parameter efficient approach for few-shot attribute value extraction. We only need to store the two sets of learned prompts with the parameters in classification head, and re-use the copy of the pre-trained Transformer backbone (Yang et al., 2022), which significantly reduces the storage cost and improves the training speed of fine-tuning. In our implementation, the backbone T5 encoder has 110M parameters and d = 768. For 24 textual prompts, 12 key prompts and 12 value prompts, they need additional 12 x (24 + 12 + 12) x 768 = 0.442M parameters. The classification matrix has 768 x 4 = 0.003M parameters. Therefore, the total number of trainable parameters in MixPAVE is 0.445M, amounting to only 0.4% of all the parameters.

4 Experiments

4.1 Datasets

We evaluate our model on two product benchmarks, AE-110K (Xu et al., 2019) and MAVE (Yang et al., 2022).

AE-110K[1] is collected from AliExpress Sports & Entertainment category, which contains over 110K data examples, i.e., product triples of {context, attribute, value}, with more than 2.7K unique attributes and 10K unique values. We select three attributes with relatively low occurrences, 'Lenses Color', 'Wheel Material' and 'Product Type', and treat them as new attributes in our experiments. 10 examples from each attribute are randomly selected as few-shot training examples.

MAVE[2] is a large and diverse dataset for product attribute extraction study, which contains 3 million attribute value annotations across 1257 fine-grained categories created from 2.2 million cleaned Amazon product profiles (Ni et al., 2019). In our experiments, we select four attributes as few-shot attributes, including 'Device Type', 'Boot Style', 'Resolution', and 'Compatibility'. We randomly select 100 examples in each attribute for fine-tuning. All other attributes are used in training the backbone. The details on the datasets are provided in Table 1.

[1] https://raw.githubusercontent.com/lanmanok/ACL19_Scaling_Up_Open_Tagging/master/publish_data.txt

[2] https://github.com/google-research-datasets/MAVE

Models	Paras	AE-110K			MAVE			
		Lenses Color	Wheel Material	Product Type	Device Type	Boot Style	Resolution	Compatibility
MAVEQA-FT (Yang et al., 2022)	100%	79.58 ± 0.39	85.59 ± 0.45	**92.41 ± 0.36**	91.06 ± 0.54	**92.23 ± 0.51**	90.88 ± 0.47	95.19 ± 0.42
Partial-1 (Yosinski et al., 2014)	8.21%	72.36 ± 0.44	79.93 ± 0.47	85.38 ± 0.62	83.69 ± 0.78	84.31 ± 0.65	84.15 ± 0.73	85.57 ± 0.59
Adapter (Pfeiffer et al., 2020)	2.53%	74.61 ± 0.55	81.84 ± 0.39	86.78 ± 0.56	85.51 ± 0.55	86.12 ± 0.47	83.22 ± 0.57	87.56 ± 0.48
BitFit (Zaken et al., 2022)	2.04%	74.95 ± 0.41	82.37 ± 0.38	87.44 ± 0.48	87.34 ± 0.58	87.67 ± 0.71	85.56 ± 0.43	88.30 ± 0.54
Prompt-Tuning (Lester et al., 2021)	0.40%	81.32 ± 0.46	84.69 ± 0.35	90.67 ± 0.40	91.43 ± 0.46	90.36 ± 0.35	92.37 ± 0.47	95.25 ± 0.41
Prefix-Tuning (Li and Liang, 2021)	0.40%	81.12 ± 0.42	84.95 ± 0.44	90.30 ± 0.45	91.79 ± 0.42	89.56 ± 0.51	92.08 ± 0.43	94.86 ± 0.45
XPrompt (Ma et al., 2022)	0.33%	82.14 ± 0.32	**85.72 ± 0.48**	90.50 ± 0.33	91.34 ± 0.48	90.67 ± 0.52	92.31 ± 0.46	95.10 ± 0.44
MixPAVE	**0.40%**	**82.58 ± 0.53**	85.36 ± 0.42	91.63 ± 0.38	**92.51 ± 0.44**	91.75 ± 0.53	**93.47 ± 0.42**	**96.86 ± 0.39**

Table 2: Performance comparison results with standard deviation on all few-shot attributes. MAVEQA-FT denotes full fine-tune of the model. "Paras" represents the number of trainable parameters in each method. Prompt length is set to 24 for both textual and key-value prompts.

4.2 Baselines

Our model is compared with seven state-of-the-art baselines, including full fine-tune over the backbone MAVEQA (Yang et al., 2022), three parameter-efficient partial tuning methods, Partial-1 (Yosinski et al., 2014), BitFit (Zaken et al., 2022) and Adapter (Pfeiffer et al., 2020), and three prompt tuning methods, Prompt-Tuning (Lester et al., 2021), Prefix-Tuning (Li and Liang, 2021) and XPrompt (Ma et al., 2022). For Partial-1, we only fine-tune the top layer of the backbone. For Prompt-Tuning and Prefix-Tuning, we use 48 prompt tokens to ensure the same number of tunable parameters.

4.3 Settings

MixPAVE is implemented using PyTorch, and is trained on 64 NVIDIA Tesla V100 GPUs. During training, we use the gradient descent algorithm with Adam (Kingma and Ba, 2015) optimizer. The backbone uses a T5-base encoder with 12 layers and 12 heads. The embedding size is 768. The maximal input sequence lengths are set to 128 and 1024 for AE-110K and MAVE dataset respectively, since the product context in MAVE has large length. The lengths of the textual, key and value prompts are 24, 12 and 12 respectively by default. We fine-tune 1k steps, with constant learning rate $1e^{-2}$ and batch size 128. Following previous works, we use F1 score as evaluation metrics and use Exact Match (Rajpurkar et al., 2016) criteria to compute the scores. Each experiment is repeated 5 times and average scores are reported.

5 Results

5.1 Main Results

We compare our MixPAVE with several state-of-the-art methods on the two product benchmarks.

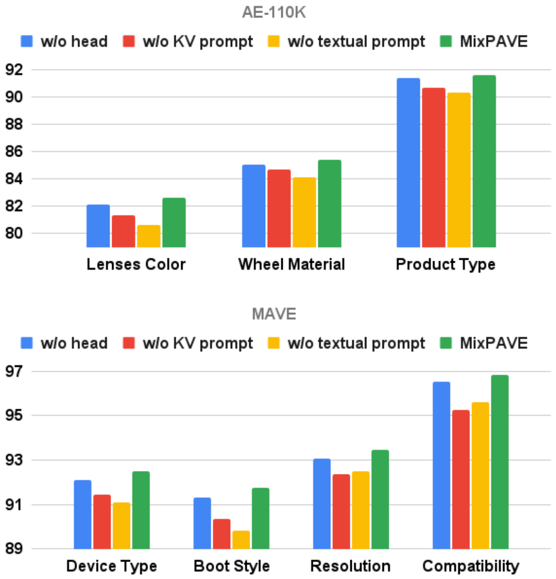


Figure 3: Ablation study on the impact of different trainable components on both datasets.

The performance comparison results are reported in Table 2. There are several key observations from these results. **First**, MixPAVE is able to catch up with the full fine-tuned backbone model (i.e., MAVEQA-FT) and even achieves better performances on certain attributes, e.g., Lenses Color and Device Type. This observation demonstrates the effectiveness of our approach for few-shot attribute value extraction. On the other hand, our model only trains 0.4% parameters in the backbone, which is much more parameter efficient than the full fine-tuned model. **Second**, it is not surprising to see that the prompt tuning based approaches generally outperform the other parameter efficient methods, such as partial fine-tuning (Partial-1) and bias fine-tuning (BitFit), indicating the superior adaptability of prompt tuning methods on large scale language models. Again, the number of tunable parameters in prompt tuning methods is also smaller compared

Length	Device Type	Boot Style	Resolution	Compatibility
4	80.31	**94.27**	93.15	96.25
8	86.84	93.16	93.24	95.89
12	90.63	92.29	93.12	96.34
24	92.51	91.75	**93.47**	96.86
48	**93.73**	92.20	93.22	96.98
96	92.86	92.86	93.41	**97.14**

Table 3: Performance comparison with different prompt lengths on MAVE dataset.

Position	Device Type	Boot Style	Resolution	Compatibility
All	**92.51**	**91.75**	**93.47**	**96.86**
Input	78.56	85.48	90.87	92.61
Bottom 6	88.35	88.64	92.70	94.17
Output	75.82	84.37	87.74	88.16
Top 6	82.55	88.49	91.52	92.63
Alternative	90.31	90.14	92.88	94.93

Table 4: Performance comparison with different prompt positions on MAVE dataset.

to the other methods. **Third**, our approach achieves the best performance among those prompt tuning methods in most cases, demonstrating the effective design of the mixed prompts. For example, the F1 score of MixPAVE increases over 1.76% and 1.61% compared with XPrompt and Prompt-Tuning, respectively, on the 'Product Type' attribute. These existing prompt tuning methods only focus on design input prompt tokens, which fail to capture the accurate interactions between tokens in the new data. In contrast, the key-value prompts in MixPAVE effectively bridge this gap.

6 Analysis and Discussion

To better understand the effectiveness of MixPAVE, we further conduct a series of ablation studies.

6.1 Impact of Different Trainable Modules

To understand the impact of different trainable components in our model, i.e., textual prompts, key-value prompts and extraction head, we conduct an ablation study by removing each component from MixPAVE individually. Concretely, removing textual prompts or key-value prompts means not adding these prompts to the model. Removing the extraction head essentially means freezing its parameters during fine-tuning. The results of F1 scores on all attributes are illustrated in Figure 3. It can be seen that the model performances drop when removing any of the trainable modules, which is consistent with our expectation. Moreover, we observe that both textual prompts and key-value prompts are crucial in few-shot extraction. For example, the F1 score decreases 1.8% and 2.1% when removing the key-value prompts and textual prompts respectively on 'Boot Style', validating the importance of these two prompts in the few-shot attribute value extraction task.

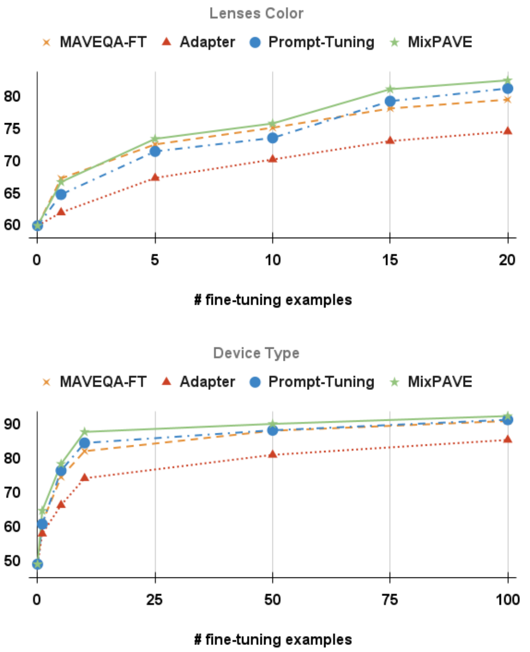


Figure 4: Performance comparison of four models with different numbers of fine-tuning examples. We select 'Lenses Color' from AE-110K and 'Device Type' from MAVE for illustration.

6.2 Impact of Prompt Length

Prompt length is the only hyper-parameter needed to tune in MixPAVE. To further analyze the impact of different prompt lengths on different attributes, we conduct another ablation study on the prompt length by modifying the prompt length from a set of values $\{4, 8, 12, 24, 48, 96\}$. Note that we simultaneously adjust the lengths of both textual prompts and key-value prompts. More discussion on how to balance these two prompts will be provided in later experiments. The model performance results on different prompt lengths are reported in Table 3. It is clear that there is no universal optimal prompt length that can achieve the best performance across all attributes. For example, on 'Boot Style', MixPAVE with prompt length 4 obtains the highest F1 score, while our model with prompt length 96 gains the best performance on 'Compatibility'. Our hypothesis is that different attributes contain different data distributions, where attribute value extraction is more difficult on certain attributes than others. These "hard" attributes usually require longer

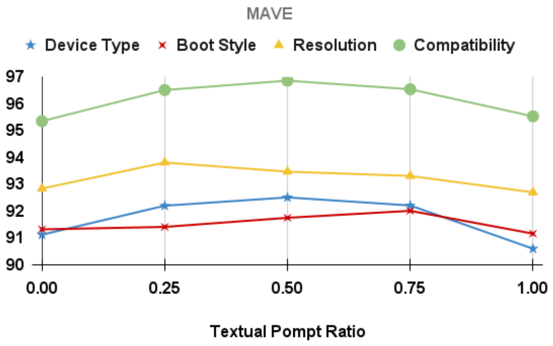


Figure 5: Performance comparison of different mixes of the two prompts on MAVE dataset.

prompts in order to better capture the patterns and knowledge from the data, with the cost of more trainable parameters.

6.3 Impact of Different Prompt Positions

In this ablation study, we evaluate the impact of different prompt positions to the model performance. Concretely, we train five additional models with different prompt locations, including only input layer, output layer, top 6 layers, bottom 6 layers and alternative layers (i.e., layers 1, 3, 5, 7, 9 and 11). The performance comparison results on MAVE are reported in Table 4. It is not surprising to see that inserting prompts to all encoder layers achieves the best performance. We can also observe that only putting the prompts to the input or output layer results in large performance drops, which is consistent with the observations in other prompt tuning works (Jia et al., 2022; Ma et al., 2022).

6.4 Impact of Fine-tuning Data Size

To further understand the model behaviors on few-shot extraction, we conduct another set of experiments on two attributes by varying the number of fine-tuning examples. Specifically, for 'Lenses Color', we evaluate our model performance with $\{0, 1, 5, 10, 15, 20\}$ annotations. For 'Device Type', we vary the number of fine-tuning examples from $\{0, 1, 5, 10, 50, 100\}$. We show the few-shot extraction results of four different models in Figure 4. There are two main observations. First, we see that our approach consistently outperforms the full fine-tuned backbone under different few-shot settings on both attributes, which further validates the effectiveness of our model. Second, when there are no annotations (equivalent to zero-shot extraction), the backbone model does not perform well. However, the performances of all compared models dramatically improve with a few fine-tuning examples, and then saturate at a certain point.

Model	Device Type	Boot Style	Resolution	Compatibility
Encoder-2L	91.45	92.12	92.96	96.49
Encoder-6L	91.82	**92.13**	93.35	**97.11**
Encoder-12L	92.51	91.75	**93.47**	96.86
Encoder-24L	**92.87**	89.78	92.85	96.41

Table 5: MixPAVE performances with different backbones on MAVE dataset.

6.5 Effect of Different Mixing Strategies

There are two sets of prompts in MixPAVE, which contribute differently to improve the model performance. To further investigate their correlation and effectiveness, we conduct an experiment by fixing the total number of trainable parameters, and adjusting the ratio of textual prompts from $\{0, 0.25, 0.5, 0.75, 1\}$. Note that 0.5 is our default setting (24 textual prompts with 12 key prompts and 12 value prompts). The model performances at different ratios on MAVE are illustrated in Figure 5. We observe slightly different patterns on different attributes. For example, on 'Resolution', textual prompts with 0.25 ratio achieves the best F1 score, while textual prompts with 0.75 ratio gives the best performance on 'Boot Style'. Nevertheless, MixPAVE with ratio 0 (no textual prompts) or 1 (no key-value prompts) underperforms other prompt combinations, indicating the effectiveness of the prompt mixing strategy.

6.6 Impact of Different Backbone Scales

We conduct a performance-scale study on different model configurations of the backbone. In particular, our base model uses a 12-layer encoder. We evaluate the model performance with a different number of encoder layers in $\{2\text{L}, 6\text{L}, 12\text{L}, 24\text{L}\}$. The F1 scores of different models on MAVE are reported in Table 5. It is interesting to see that Encoder-24L does not always yield the best performance on all attributes. This observation is consistent with the experimental results in Table 3. The reason is that large models or models with large trainable parameters might overfit to certain attributes, especially in few-shot settings, resulting in worse model performances.

7 Conclusions

Product attribute value extraction on new attributes is an important problem in many real-world applications. In this work, we propose a novel prompt tuning approach with Mixed Prompts for few-shot Attribute Value Extraction (MixPAVE). In particular, our model introduces only a small amount of trainable parameters, consisting of two sets of learnable prompts, while keeping the backbone extraction model frozen. Our MixPAVE not only benefits from parameter-efficient training, but also avoids model overfitting on limited training examples. Experimental results on AE-110k and MAVE demonstrate the effectiveness and efficiency of the proposed approach.

Limitations

There are two limitations of the current MixPAVE model. First, although MixPrompt can achieve comparable extraction performance with full fine-tuning, how to identify the optimal combination of the two prompts is challenging and remains unanswered. We conduct grid search in our experiments to empirically find the best prompts length. In future, we plan to investigate a systematic solution for identifying the optimal or a suboptimal combination. Second, our model learns attribute-specific prompts for a new attribute. We plan to explore a parametric network that could guide the learning of attribute-agnostic prompts.

References

Julian Brooke, Adam Hammond, and Timothy Baldwin. 2016. Bootstrapped text-level named entity recognition for literature. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*. The Association for Computer Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Han Cai, Chuang Gan, Ligeng Zhu, and Song Han. 2020. Tinytl: Reduce memory, not parameters for efficient on-device learning. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Ke Chen, Lei Feng, Qingkuang Chen, Gang Chen, and Lidan Shou. 2019. EXACT: attributed entity extraction by annotating texts. In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2019, Paris, France, July 21-25, 2019*, pages 1349–1352. ACM.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Vishrawas Gopalakrishnan, Suresh Parthasarathy Iyengar, Amit Madaan, Rajeev Rastogi, and Srinivasan H. Sengamedu. 2012. Matching product titles using web-based enrichment. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*, pages 605–614. ACM.

Demi Guo, Alexander M. Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4884–4896. Association for Computational Linguistics.

Junxian He, Chunting Zhou, Xuezhe Ma, Taylor Berg-Kirkpatrick, and Graham Neubig. 2022a. Towards a unified view of parameter-efficient transfer learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross B. Girshick. 2022b. Masked autoencoders are scalable vision learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 15979–15988. IEEE.

Yun He, Huaixiu Steven Zheng, Yi Tay, Jai Prakash Gupta, Yu Du, Vamsi Aribandi, Zhe Zhao, YaGuang Li, Zhao Chen, Donald Metzler, Heng-Tze Cheng, and Ed H. Chi. 2022c. Hyperprompt: Prompt-based task-conditioning of transformers. In *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pages 8678–8690. PMLR.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Ronghang Hu, Amanpreet Singh, Trevor Darrell, and Marcus Rohrbach. 2020. Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 9989–9999. Computer Vision Foundation / IEEE.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR*, abs/1508.01991.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge J. Belongie, Bharath Hariharan, and Ser-Nam Lim. 2022. Visual prompt tuning. In *Computer Vision - ECCV 2022 - 17th European Conference, Tel Aviv, Israel, October 23-27, 2022, Proceedings, Part XXXIII*, volume 13693 of *Lecture Notes in Computer Science*, pages 709–727. Springer.

Giannis Karamanolakis, Jun Ma, and Xin Luna Dong. 2020. Txtract: Taxonomy-aware knowledge extraction for thousands of product categories. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8489–8502. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4582–4597. Association for Computational Linguistics.

Rongmei Lin, Xiang He, Jie Feng, Nasser Zalmout, Yan Liang, Li Xiong, and Xin Luna Dong. 2021. PAM: understanding product images in cross product category attribute extraction. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 3262–3270. ACM.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pretrain, prompt, and predict: A systematic survey of prompting methods in natural language processing. *CoRR*, abs/2107.13586.

Hanqing Lu, Youna Hu, Tong Zhao, Tony Wu, Yiwei Song, and Bing Yin. 2021. Graph-based multilingual product retrieval in e-commerce search. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 146–153. Association for Computational Linguistics.

Fang Ma, Chen Zhang, Lei Ren, Jingang Wang, Qifan Wang, Wei Wu, Xiaojun Quan, and Dawei Song. 2022. Xprompt: Exploring the extreme of prompt tuning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates*, page 11033–11047. Association for Computational Linguistics.

Thanh V. Nguyen, Nikhil Rao, and Karthik Subbian. 2020. Learning robust models for e-commerce product search. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 6861–6869. Association for Computational Linguistics.

Jianmo Ni, Jiacheng Li, and Julian J. McAuley. 2019. Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 188–197. Association for Computational Linguistics.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 487–503. Association for Computational Linguistics.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulic, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos, Online, November 16-20, 2020*, pages 46–54. Association for Computational Linguistics.

Duangmanee Putthividhya and Junling Hu. 2011a. Bootstrapped named entity recognition for product attribute extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1557–1567. ACL.

Duangmanee Putthividhya and Junling Hu. 2011b. Bootstrapped named entity recognition for product attribute extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1557–1567. ACL.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21:140:1–140:67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 2383–2392. The Association for Computational Linguistics.

Ohad Rozen, David Carmel, Avihai Mejer, Vitaly Mirkis, and Yftah Ziser. 2021. Answering product-questions by utilizing questions from other contextually similar products. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021*, pages 242–253. Association for Computational Linguistics.

Keiji Shinzato, Naoki Yoshinaga, Yandi Xia, and Wei-Te Chen. 2022. Simple and effective knowledge-driven query expansion for qa-based product attribute extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 227–234. Association for Computational Linguistics.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards VQA models that can read. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 8317–8326. Computer Vision Foundation / IEEE.

Hao Tan and Mohit Bansal. 2019. LXMERT: learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*, pages 5099–5110. Association for Computational Linguistics.

Quoc-Tuan Truong, Tong Zhao, Changhe Yuan, Jin Li, Jim Chan, Soo-Min Pantel, and Hady W. Lauw. 2022. Ampsum: Adaptive multiple-product summarization towards improving recommendation captions. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 2978–2988. ACM.

Damir Vandic, Jan-Willem van Dam, and Flavius Frasincar. 2012. Faceted product search powered by the semantic web. *Decis. Support Syst.*, 53(3):425–437.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Qifan Wang, Li Yang, Bhargav Kanagal, Sumit Sanghai, D. Sivakumar, Bin Shu, Zac Yu, and Jon Elsas. 2020. Learning to extract attribute value from product via question answering: A multi-task approach. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*, pages 47–55. ACM.

Qifan Wang, Li Yang, Jingang Wang, Jitin Krishnan, Bo Dai, Sinong Wang, Zenglin Xu, Madian Khabsa, and Hao Ma. 2022. Smartave: Structured multimodal transformer for product attribute value extraction. In *In Findings of the Association for Computational Linguistics: EMNLP 2022, Abu Dhabi, United Arab Emirates.*, page 263–276. Association for Computational Linguistics.

Huimin Xu, Wenting Wang, Xin Mao, Xinyu Jiang, and Man Lan. 2019. Scaling up open tagging from tens to thousands: Comprehension empowered attribute value extraction from product title. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5214–5223. Association for Computational Linguistics.

Jun Yan, Nasser Zalmout, Yan Liang, Christan Grant, Xiang Ren, and Xin Luna Dong. 2021. Adatag: Multi-attribute value extraction from product profiles with adaptive decoding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 4694–4705. Association for Computational Linguistics.

Li Yang, Qifan Wang, Zac Yu, Anand Kulkarni, Sumit Sanghai, Bin Shu, Jon Elsas, and Bhargav Kanagal. 2022. MAVE: A product dataset for multi-source attribute value extraction. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining, Virtual Event / Tempe, AZ, USA, February 21 - 25, 2022*, pages 1256–1265. ACM.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3320–3328.

Sanshi Yu, Zhuoxuan Jiang, Dongdong Chen, Shanshan Feng, Dongsheng Li, Qi Liu, and Jinfeng Yi. 2021. Leveraging tripartite interaction information from live stream e-commerce for improving product recommendation. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021*, pages 3886–3894. ACM.

Elad Ben Zaken, Yoav Goldberg, and Shauli Ravfogel. 2022. Bitfit: Simple parameter-efficient fine-tuning for transformer-based masked language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 1–9. Association for Computational Linguistics.

Jeffrey O. Zhang, Alexander Sax, Amir Zamir, Leonidas J. Guibas, and Jitendra Malik. 2020a. Side-tuning: A baseline for network adaptation via additive side networks. In *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part III*, volume 12348 of *Lecture Notes in Computer Science*, pages 698–714. Springer.

Wenxuan Zhang, Yang Deng, Jing Ma, and Wai Lam. 2020b. Answerfact: Fact checking in product question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2407–2417. Association for Computational Linguistics.

Xinyang Zhang, Chenwei Zhang, Xian Li, Xin Luna Dong, Jingbo Shang, Christos Faloutsos, and Jiawei Han. 2022. Oa-mine: Open-world attribute mining for e-commerce products with weak supervision. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*, pages 3153–3161. ACM.

Guineng Zheng, Subhabrata Mukherjee, Xin Luna Dong, and Feifei Li. 2018. Opentag: Open attribute value extraction from product profiles. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD 2018, London, UK, August 19-23, 2018*, pages 1049–1058. ACM.

Tiangang Zhu, Yue Wang, Haoran Li, Youzheng Wu, Xiaodong He, and Bowen Zhou. 2020. Multimodal joint attribute prediction and value extraction for e-commerce product. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 2129–2139. Association for Computational Linguistics.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
Left blank.

□ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Left blank.

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

□ B1. Did you cite the creators of artifacts you used?
Not applicable. Left blank.

□ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

□ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

□ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

□ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

□ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Left blank.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Left blank.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Left blank.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Left blank.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

□ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

□ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

□ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

□ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

□ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.