# Pay Attention to Implicit Attribute Values: A Multi-modal Generative Framework for AVE Task

**Yupeng Zhang[1]*, Shensi Wang[2]*, Peiguang Li[2], Guanting Dong[4], Sirui Wang[2,3]†, Yunsen Xian[2], Zhoujun Li[1] and Hongzhi Zhang[2]**

[1]Beihang University, Beijing China [2]Meituan Inc., Beijing China
[3]Department of Automation, Tsinghua University, Beijing China
[4]Beijing University of Posts and Telecommunications, Beijing China
{G0vi_qyx, lizj}@buaa.edu.cn {dongguanting}@bupt.edu.cn
{wangshensi02, lipeiguang, wangsirui}@meituan.com
{xianyunsen, zhanghongzhi03}@meituan.com

## Abstract

Attribute Value Extraction (AVE) boosts many e-commerce platform services such as targeted recommendation, product retrieval and question answering. Most previous studies adopt an extractive framework such as named entity recognition (NER) to capture subtokens in the product descriptions as the corresponding values of target attributes. However, in the real world scenario, there also exist implicit attribute values that are not mentioned explicitly but embedded in the image information and implied text meaning of products, for which the power of extractive methods is severely constrained. To address the above issues, we exploit a unified multi-modal AVE framework named DEFLATE (a multi-modal unifie**D** fram**E**work **F**or imp**L**icit **A**nd explici**T** AV**E**) to acquire implicit attribute values in addition to the explicit ones. DEFLATE consists of a QA-based generation model to produce candidate attribute values from the product information of different modalities, and a discriminative model to ensure the credibility of the generated answers. Meanwhile, to provide a testbed that close to the real world, we collect and annotate a multi-modal dataset with parts of implicit attribute values. Extensive experiments conducted on multiple datasets demonstrate that DEFLATE significantly outperforms previous methods on the extraction of implicit attribute values, while achieving comparable performances for the explicit ones.

## 1 Introduction

A wide range of e-commerce platforms benefit from accurate annotated product attributes and their values, which could facilitate customers to understand the product better as additional information and help users search for preferred products as a key word (Cao et al., 2018). However, it is a pervasive phenomenon that the manually annotated attribute values of most products on the e-commerce platforms are incomplete and noisy, due to the tedious nature of this work (Dong et al., 2020). To address this, many researches have been proposed for the task of Attribute Value Extraction (AVE), aiming at extracting the values of the attributes from the product information such as title or description.

Current mainstream methods on AVE treat it as an information extraction task, the representative models include NER and machine reading comprehension (MRC). The former adopt a set of entity tags for each attribute (e.g., "B-Color" and "I-Color" for the attribute "Color") to identify the corresponding attribute values (Chiu and Nichols, 2016; Lample et al., 2016; Zheng et al., 2018; Huang et al., 2015; Xu et al., 2019). While the latter tackles AVE by intercepting spans from the product textual information as the values of a given attribute. (Wang et al., 2020; Shinzato et al., 2022).

Existing works mainly focus on the extraction for the attribute values that are mentioned explicitly in the product descriptions. While there exists significant gap between them and the real world scenario, where an attribute value that needs to be obtained does not usually appear as a subsequence of the product description, but can be inferred from the image, implied text meaning and prior knowledge. Fig. 1 shows an intuitive case of this situation, in which the value ("*8 inches*") corresponding to the attribute of "size" appears in the image in-

*Work done during internship at Meituan Inc. The first two authors have equal contributions.
†Corresponding author.



*Findings of the Association for Computational Linguistics: ACL 2023*, pages 13139–13151

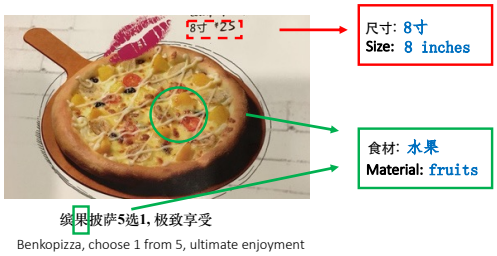


Figure 1: An example of a product with its textual and visual description. The attribute value to be extracted is not a subsequence of product description.

stead of the description, and the value ("*fruits*") of the "material" attribute is implicitly embedded in textual and visual information. So for extractive methods, it is hard to extract those implicit attribute values accurately by taking snippets from the product descriptions.

Inspired by recent progress of text generation paradigms on the field of Natural Language Processing (NLP) (Dong et al., 2023), Roy et al. (2022) explored applying generative models to tackle the AVE task. They propose word and positional sequence-based patterns to jointly generate the attributes and the corresponding values. However, this method is still designed to find explicit answers from the textual descriptions in essence, without incorporating the information of images, as well as the semantic information about the attribute names. Following this research direction, we further formulate AVE as a generative question-and-answer (QA) task, thus empowering the model to predict (generate) the implicit attribute values (IAV). Specifically, we design a multi-modal unifie**D** fram**E**work **F**or imp**L**icit **A**nd explici**T** AV**E** (called DEFLATE in brief), which takes textual and visual information of products as inputs and adopts an *Encoder-to-Decoder* framework.

In *Encoder*, taking an attribute as the query, DEFLATE combines it with the textual description and the image of a product into a single sequence, and a novel multi-modal T5 model is applied to encode the source data incorporating a cross-modal attention mechanism. In *Decoder*, all possible values of a query attribute are generated one by one as the candidate answers. Besides, to select the expected attributes, a discriminative model is introduced to determine whether a certain attribute and corresponding values belong to a product or not.

Moreover, as mentioned above, the existing public AVE datasets (such as MAVE (Yang et al., 2022) and AliExpress (Xu et al., 2019)) offer an ideal situation for the extractive methods, in which all attribute values appear explicitly in the textual information. Considering the absence of a testbed for the evaluation of implicit AVE, we present a multi-modal dataset with a considerable number of IAV to support more future related work.

Our contributions can be summarized as follows:

- We propose a unified framework for AVE task, which consists of a multi-modal attribute generative model incorporating visual information to generate both explicit and implicit attribute values, and a discriminative model to filter the generated answers.
- We present a challenging dataset including texts and images to support researches on the extraction of implicit attribute values. To the best of our knowledge, we are the first to concentrate on the extraction of implicit attribute values.[1]
- Extensive experiments show that our method outperforms previous works sharply for the extraction of the products with IAV, and are also well-performed for extracting explicit attribute values.

## 2 Related Work

### 2.1 Extractive Methods for AVE

Early AVE works are mainly rule-based approaches that design regular expressions to recognize phrases that indicate the values of every attribute using human-crafted domain-specific seed dictionary (Vandic et al., 2012; Gopalakrishnan et al., 2012). With the development of deep neural networks, various deep learning-based methods have achieved success on the sequence tagging task, which is similar to NER (Chiu and Nichols, 2016; Lample et al., 2016). For example, models based on BiLSTM-CRF are applied to a sequence tagging task successfully (Kozareva et al., 2016; Huang et al., 2015), and some extended methods like LSTM-CNNs-CRF model have also achieved significant performances on this task (Ma and Hovy, 2016). Furthermore, recent works regard AVE as a sequence tagging task, in which values for each attribute $a$

[1]Our dataset and code are publicly available at https://github.com/G0vi/DEFLATE

in a sentence will be tagged as "B-$a$" and "I-$a$", denoting beginning and inside of subsequences attributed to $a$ (Karamanolakis et al., 2020; Zheng et al., 2018). To improve the scalability of models for large set of attributes and unseen attributes, Xu et al. (2019) consider each attribute as a query added to the attention layer of the product title, and adopt a global BIO tags for all the attributes.

There are also some NER models incorporating multi-modal product information for the AVE task (Zhu et al., 2020). Besides, other methods also try to formulate AVE as a QA task in MRC, with the goal to extract spans for a given attribute (Wang et al., 2020; Shinzato et al., 2022). Both sequence tagging and MRC methods above are extractive methods, which are only applicable to the situations when the attribute values completely appear in the product descriptions, but can hardly extract those attributes embedded in the information of contexts or images implicitly without directly mentioned in the texts.

### 2.2 Generative Methods for AVE

Motivated by successful works such as T5 (Raffel et al., 2020) that use generative techniques as a unified solution for various NLP tasks including text classification and slot filling, several researchers have also formulated AVE as a text generation task. Roy et al. (2021) combine the context and an attribute with its value masked as blank, and utilize Infilling by Language Modeling (ILM) (Donahue et al., 2020) to generate the missing span as the prediction of the value. Roy et al. (2022) propose two generative paradigms: word sequence-based and positional sequence-based to tackle the AVE task, which jointly generate the values and positions in the text with their corresponding attribute names one by one. But generating values and their attributes by positions still needs the attribute values to appear explicitly in the product titles or descriptions. And both the two methods above have not considered multi-modal information of the products. Cho et al. (2021) unify vision and language tasks using a multi-modal text generation framework. And analogously in the field of AVE, Lin et al. (2021) tackle the problem by a seq2seq model called PAM that combines the product texts, Optical Character Recognition (OCR) tokens and visual objects detected in the product image in the encoder, and the decoded tokens are selected from the above inputs as well as a dynamic vocabulary of values. However, PAM just uses detected OCR tokens and partial objects in an image by pretrained models, which could lose some visual information that is unimportant to the pretraining task but useful for AVE. Besides, it only focuses on value extraction for given attributes, without considering which attributes should appear in the outputs. In addition, Tavanaei et al. (2022) propose MMT4, a transformer framework with multi modality to improve the performance of generative models in e-commerce. It is also a vision-projected sequence-to-sequence architecture with image feature vectors processed by ViT or CNN. Compared to MMT4, our DEFLATE incorporates a discriminative model to enhance the quality of generated samples, and additionally utilizes a more lightweight image encoding scheme with less information loss of images than ViT and CNN.

## 3 Approach

In this paper, a new framework DEFLATE is introduced for attribute value extraction. We unify the extraction for both explicit and implicit attribute values of products as a multi-modal text generation task. The main idea of DEFLATE is to generate candidate values for each attribute conditioned on the attribute name and the product information first, and then determine the values of which attributes pertain to a certain product using a discriminative model. The overall architecture of DEFLATE is visualised in Fig. 2.

### 3.1 Text Embedding

To begin with, we design a uniform QA format for training. The text information of each product $p$ includes its description $des_p$ and name $name_p$. For this task, we combine all textual information above of a product as well as an attribute $A$ into an question sequence as {The [$A$] of [$name_p$] ([$des_p$]) are}. We encode it as the text embedding $e_t = \{t_1, t_2, ..., t_n\} \in \mathbb{R}^{n \times d}$, where $n$ is the sequence length and $d$ is the hidden dimension of transformer. The encoder, decoder and language modeling head use a shared group of embedding parameters. Following T5, we adopt relative position embedding and add relative position bias to each self-attention layer.

### 3.2 Image Embedding

Directly feeding the high resolution image pixels to the models requires excessive memory and time

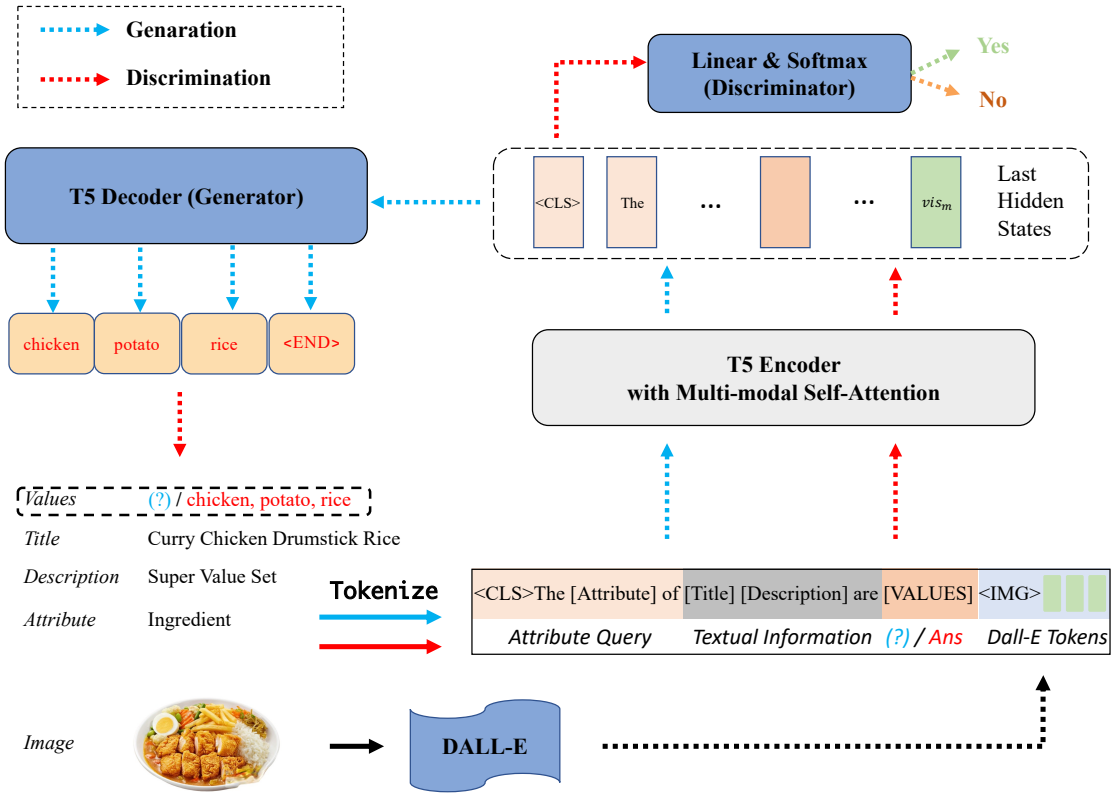


Figure 2: The generative-and-discriminative modules of DEFLATE for the AVE task. The generator and discriminator use a shared encoder. The blue dashed line represents the generation process, and the red is for the discriminant module.

for training. For more lightweight training and inference, we compress the images using DALL-E (Ramesh et al., 2021), a technique based on discrete variational auto-encoder. It converts each 256×256 RGB image into 32×32 tokens, which can assume 8192 possible values for each element.

The visual tokens are denoted as $V = \{[vis_1], [vis_2], \cdots, [vis_m]\}$, where $m$ is the number of tokens. We represent each token $[vis_i]$ as a one-hot vector. The corresponding visual embedding of each image can be obtained as follows:

$$e_v = VW = \{v_1, v_2, ..., v_m\} \in \mathbb{R}^{m \times d}, \quad (1)$$

where $W \in \mathbb{R}^{8192 \times d}$ is a trainable matrix in the same latent $d$-dim space with the texts, and $v_i$ is the embedding of $[vis_i]$.

### 3.3 Encoder-Decoder Architecture

Our multi-modal generative model overall follows the encoder-decoder architecture of T5. The multi-modal T5 (Mul-T5) encoder consisting of 12 transformer blocks jointly encodes the concatenated text and image tokens of the products. Specifically, we add a special token <IMG> to separate the textual and visual tokens. Thus the multi-modal feature fed into the feed-forward network of the encoder is

$$\begin{aligned} h &= Enc(e_t, e_v) \\ &= \{t_1, t_2, \cdots, t_n, h_{\text{IMG}}, v_1, v_2, \cdots, v_m\}, \end{aligned} \quad (2)$$

where $h_{\text{IMG}}$ is the embedding of the token <IMG>. Similar to the encoder, the decoder also has 12 transformer blocks with an additional cross-attention layer for each. The decoder generates text sequences conditioned on the hidden features of the encoder and the previously generated tokens. As with all previously used seq2seq models, the architecture is trained to maximize the likelihood over each token of the value sequence: $P_\theta(y_j | y_{1:j-1}, h)$, where $\theta$ represents the parameters of the generator and $y_j$ is the $j$-th token of the value sequence.

### 3.4 Attribute Discriminative Module

The QA-based generator is designed to complete the values of the given attribute, but actually not

every product has all attributes. For example, in our QA approach, a question like "*What is the cream type of braised pork?*" would arise, but indeed the "*cream type*" is hardly related to "*braised pork*". Therefore, we introduce an attribute discrimination module (discriminator in brief) with a shared encoder with generator for attribute prediction to restrict the results of the generator. The architecture of the discriminator is shown in Fig. 2. To be specific, for a certain product $p$, given an attribute name $a_i$, the product's text tokens $T_p$, image tokens $V_p$ extracted by DALL-E and the corresponding attribute values $v_{p,i}$ generated by generator, the input sequence fed into discriminator can be expressed as $s_{p,i}$ ={<CLS> The [$a_i$] of [$T_p$] is [$v_{p,i}$] <IMG> $V$}, where <CLS> is a special classification token that appears in the first position of every input sequence. The final hidden embedding $C$ of <CLS> serves as the aggregate sequence representation for classification. The score function for the input sequence $s_{p,i}$ is

$$D(s_{p,i}) = \text{sigmoid}(\text{MLP}(C)). \quad (3)$$

The training criterion for the discriminator is to minimize the cross-entropy loss:

$$L(p,i) = -\sum_{s_{p,i} \in S^+ \cap S^-} (y_{p,i} \log(D(s_{p,i})) + (1 - y_{p,i}) \log(1 - D(s_{p,i}))), \quad (4)$$

where $S^+$ and $S^-$ denote positive and negative samples respectively, and $y_{p,i} = 1$ when $s_{p,i} \in S^+$ and $y_{p,i} = 0$ when $s_{p,i} \in S^-$. Specifically, whether a sequence is a positive or negative sample depends on if the attribute $a_i \in A_p$ which refers to the attribute set of $p$. We take each attribute $a_{i+} \in A_p$ with its values $v_{p,i}$ in the ground truth label of $p$ for the positive samples, and sample some attributes that satisfy $a_{i-} \notin A_p$ for the negative ones. As for the values $v_{p,i}$ of each $a_{i-}$ in the negative sample sequences, we employ a generative sampling strategy. During the training of generator, we fetch the generated results for each $a_{i-}$ as $v_{p,i}$ and combine it into the product information as a negative sample sequence $s_{p,i} \in S^-$. As an example, *"What is the size of red-cooked pork?"* serves as an input to the generative model and the output result is *"10 inches"*. Then *"<CLS> The size of red-cooked pork is 10 inches"* can be used as a negative sample. According to this scheme, we construct a total of nearly 100,000 training samples. After training, we can use the discriminator to determine the reasonableness of the results given by the generative model and then output the final filtered attributes and values.

## 4 Dataset

| Attribute | Candidacy |
|---|---|
| Main material | 3670 |
| Accessories | 2048 |
| Cooking methods | 35 |
| Meat or vegetable | 3 |
| Sweet or not | 2 |
| Cool or hot | 9 |
| Cream type | 6 |
| Tea | 127 |
| hot | 10 |

Table 1: Statistics of the our dataset DESIRE.

In this paper, we propose DESIRE (multi-mo**D**al gourm**E**t product**S** with **I**mplicit att**R**ibute valu**E**s), a dataset in the gastronomy domain. We collect the data from a large e-commerce platform in China and have been licensed to release it for research purposes. DESIRE contains corresponding textual information and image information for each product. Meanwhile, as a dataset supporting the AVE task, we provide ten candidate product attributes: "Main Ingredients", "Accessories", "Flavor", "Cooking Method", etc. The size of candidate words for each attribute are shown in Table 1. The annotated attribute values of each product are labeled according to its detail information provided by merchants. And in order to prevent merchants from missing certain product attributes, annotators with extensive experience in the gourmet e-commerce field further check the attribute information for the products in the test set.

Besides, in fact, merchants do not always provide a detailed text description for each product, and the attribute values of products are not all included in the text information. Therefore, to be more fit with the real-world scenario, we directly use the real text information provided by the merchant. More specifically, each product is guaranteed to have a corresponding short text, but only about 27.83% of the products have long text descriptions.

## 5 Experiments

### 5.1 Experimental Settings

**Baselines** We conduct experiments on DESIRE and evaluate the performances of models for both attribute prediction (AP) and attribute value extraction (AVE). And to make the evaluation reliable and reasonable, the following baselines involving both extractive and generative models are selected for comparison due to their reported superior results.

- **BiLSTM-CRF** (Kozareva et al., 2016) is a general baseline for NER with a BiLSTM-based encoder to capture the semantic feature and a CRF-based decoder to calculate the maximum probability label corresponding to each token in the input sequence.
- **M-JAVE** (Zhu et al., 2020) is a multi-modal NER framework and labels the input textual product descriptions as "BIO" sequences. It feeds multi-modal features to a regional-gated cross-modality attention layer, and jointly makes the attribute prediction.
- **Jointly Generative AVE** (Roy et al., 2022) tackles both AP and AVE jointly in a generative manner. We abbreviate it as JG-AVE in the following. JG-AVE proposes two paradigms: word sequence-based and positional sequence-based that generates values or value positions with the corresponding attributes one by one in the order of appearance in the product description. Since the IAV have no specific positional information, JG-AVE is still restricted to extracting only EAV due to the sequential information of the attribute values it utilizes. And in order to make full use of its ability to extract IAV, we just implement a variant of word sequence-based JG-AVE in our experiments. Different from the original method, we include all attribute values (both IAV and EAV) into the long sequence labels in a random order, regardless of where they appear.

**Evaluation Metrics** To better evaluate the performances of the model on the tasks, we compute the F1-micro score for AP and AVE separately. And it should be noted that the evaluation calculation for AVE is based on the evaluation results for AP. In other words, a value is right if and only if both the predicted attribute and the value are matched with the label. Meanwhile, the extractive methods can hardly acquire exactly the same attribute values as in the label. So to provide a better situation for the extractive methods, we also calculate the Fuzzy F1 score for which fuzzily matched predicted and labeled attribute values are counted as correct. A predicted value fuzzily matches a label when their common substring length exceeds half of the label length.

More implementation details of our experiments are shown in A.1.

### 5.2 Main Results

| Data Range | Models | Value | | Attribute |
|---|---|---|---|---|
| | | F1 | Fuzzy F1 | F1 |
| All | BiLSTM-CRF | 8.24 | 9.99 | 22.68 |
| | M-JAVE | 9.2 | 10.56 | 25.91 |
| | JG-AVE | 23.72 | 26.70 | 58.22 |
| | DEFLATE | **40.10** | **42.89** | **89.39** |
| Only EAV | BiLSTM-CRF | 29.97 | 31.01 | - |
| | M-JAVE | 31.07 | 32.96 | - |
| | JG-AVE | 48.20 | 51.60 | - |
| | DEFLATE | **53.69** | **56.89** | - |

Table 2: Main results of comparative methods and our method on our dataset DESIRE.

On our DESIRE dataset, we evaluate the performances of our model and three other baselines on both AP and AVE tasks. The main results are presented in Table 2.

Comparing to the recent extractive model M-JAVE, DEFLATE earns a substantial increase in metrics on both tasks, where DEFLATE achieves 31 and 73 points improvement in AVE F1 score and AP F1 score. The main reason is that the mechanism of extractive models makes it difficult to predict product attributes and values not mentioned in the text information, which could be alleviated in generative frameworks.

Another generative model JG-AVE earns a better performance than the extractive ones, but DEFLATE still has an increase of 19% value predict F1 scores and 41% attribute predict F1 scores on the basis of it. JG-AVE generates long sequences of attributes and values directly through the generative model, while our model uses the discriminative model for AP and then generates values of a given attribute each time by the QA approach. However, it could be a tough task for JG-AVE to learn the attention information in its decoder when the sequence to be generated is too long and lacks order, which is alleviated in our QA-based DEFLATE. In addition, our framework additionally utilizes image

information with more potential attributes beyond the texts, which is also one of the reasons for the better performance of DEFLATE.

The overall results show that our proposed DEFLATE outperforms other baseline models significantly on both tasks.

### 5.3 Adaptability Study

| Models | Value | | Attribute |
|---|---|---|---|
| | F1 | Fuzzy F1 | F1 |
| BiLSTM-CRF | 75.83 | 78.91 | 85.23 |
| JG-AVE | 82.07 | 85.26 | 92.53 |
| M-JAVE | 85.11 | 86.36 | 90.96 |
| DEFLATE | **86.01** | **87.12** | **96.09** |

Table 3: Comparative results on the dataset MEPAVE.

As mentioned before, DEFLATE is better-performed on both explicit and implicit attribute value extraction. In addition, for the DESIRE dataset, we also evaluate the models on the extraction for only explicit attribute values (EAV) by filtering out the attribute values provided by the labels or predicted by the models that are not mentioned in the product title or description. And to further verify the adaptability of DEFLATE, we conduct additional experiments on another multi-modal Chinese AVE dataset MEPAVE (Zhu et al., 2020) (with 71,194 instances in the train set and 8,000 instances each in the dev and test set), which is proposed simultaneously with M-JAVE. It is worth noting that all attribute values in MEPAVE are mentioned explicitly in the text descriptions. What additionally needs to be declared is that we apply for the full data of MEPAVE which is not completely open source from the creators. And our use of the dataset is as expected for AVE-related research.

Extraction of EAV is more suitable for extractive models which directly label the text sequences. While as shown in Table 2 and 3, the two generative models still have comparable performances to the extractive ones. Especially, our DEFLATE outperforms M-JAVE with 23.9% AVE F1 score for EAV on DESIRE and 5.1% AP F1 score on MEPAVE, which further demonstrates DEFLATE could handle AVE well in various scenarios.

In addition, general results of the experiments on both datasets demonstrate that AP is an easier task compared to AVE. And based on this, we make a further analysis on the comparison between the two generative methods. According to the mode of training, DEFLATE completes the AVE task with a given attribute and could generate values incorporating the semantic information of the corresponding attribute names. While JG-AVE generates a value first and then predicts the corresponding attribute of it. Both methods establish a relationship between the attributes and values, but DEFLATE predicts attributes separately and utilizes the relationship when making AVE, while the occasion for JG-AVE to make use of the relationship is in the easier task (AP) instead of the harder one (AVE), for which JG-AVE is inferior to DEFLATE to a certain extent.

### 5.4 Ablation Study

To demonstrate the effectiveness of the different modules of our framework, we have performed the ablation experiments in the two aspects, 1) removing visual information of products to the encoder of generator (the models without cross modality encoding will be marked with "$v-$"); 2) getting rid of the discriminant module (the variants of DEFLATE with only a generator will be marked with "$d-$").

| Models | Value | | Attribute |
|---|---|---|---|
| | F1 | Fuzzy F1 | F1 |
| DEFLATE$^{d-,v-}$ | 31.23 | 36.74 | 70.26 |
| DEFLATE$^{d-}$ | 34.95 | 37.52 | 72.47 |
| DEFLATE$^{v-}$ | 36.66 | 39.97 | 89.28 |
| DEFLATE | **40.10** | **42.89** | **89.39** |

Table 4: Ablation study result on the DESIRE dataset.

**Ablating the Visual Information** To validate the influence of the visual information, we exclude the image tokens from the input sequences for the visual ablation experiment.

As shown in Table 4, on the testing set, without the visual information injection, DEFLATE$^{v-}$ has a 3.44% drop in F1 score on the AVE task compared to DEFLATE. And without the discriminative model in the framework, the model considering images (DEFLATE$^{d-}$) also outperforms DEFLATE$^{d-,v-}$ that neither has a discriminative module nor utilizes visual information. In addition, from the comparison between DEFLATE and DEFLATE$^{v-}$, we further find that in the presence of a discriminant module, the trained discriminator incorporating visual information performs nearly the same as the one trained only on texts for attribute prediction, which indicates the attributes of a product could be determined just by the seman-

tic information of the product and attribute name as well as the knowledge memorized by the discriminative model. And the better performances of the multi-modal generative models on AVE further illustrates there would exist additional non-negligible information about attribute values hidden in the images of some products.

**Ablating the Discriminant Module** In the comparison experiment where the discriminator is discarded, we mark the value for those attributes not belonging to a certain product as "None" to ensure that a single generator can also make attribute prediction while extracting attribute values. In this way, given a product and an attribute, the trained generator generates regular values when it determines that the attribute belongs to the product, and generates "None" otherwise.

When we complete the two tasks with only the generator, the f1 value of the AP task in the table drops from 89.39% to 72.4%. For AP, the discrimination range of the generated model is as large as the size of the dictionary, while the labels of our discrimination model are only "0" and "1", which is relatively simpler and more accurate. Moreover, both DEFLATE$^{d-}$ and DEFLATE$^{d-,v-}$ also have degraded performances on AVE. This is probably because the token "None" that is not a real value has a different meaning from others in the generative training, which would cause a gap between the distribution of attribute values learned by the generator and that in the training data.

### 5.5 Compressed Visual Information

In this section, our study focuses on whether the visual tokens compressed by DALL-E will lose some necessary information for the AVE and AP tasks. In the comparative experiment, instead of obtaining the output embedding from the condensed DALL-E tokens, we feed the original images of the products into a ViT (Dosovitskiy et al., 2020) model and get the embedding of 16×16 patches. In our implementation, ViT-base-patch16-224 (Wu et al., 2020) is chosen as the backbone model to extract the image features.

As shown in Table 5, the model extracting features from original images does not show observably better results than that utilizing the compressed image tokens. Since the compressed tokens encoded by DALL-E could be restored to the images that look almost identical to the original ones without a large degradation in visual quality, human-visible attribute elements in the images of products are not lost. Moreover, training with the patch embedding extracted on ViT brings much more computational and memory overhead compared to the DEFLATE (DALL-E). So our method is time-efficient and resource-efficient for the injection of visual modality on AVE.

| Models | Value | Attribute |
|---|---|---|
| | F1 | F1 |
| DEFLATE (DALL-E) | 40.10 | **89.39** |
| DEFLATE (ViT) | **41.02** | 89.36 |

Table 5: Performances of models trained with full or compressed visual information on DESIRE.

### 5.6 Low-Resource Evaluation

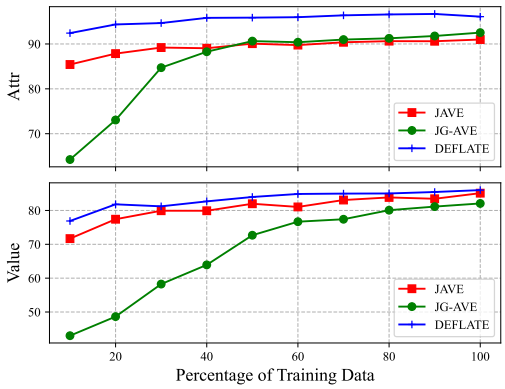


Figure 3: Performances of different methods on different proportions of the MEPAVE data.

For further investigation of our method and the baselines on small datasets, we divide the training set into 10 parts and train models with different proportions. To contrastively evaluate the performances of the extractive models with low resource, we conduct the experiments on MEPAVE. The evaluation results of models on the two subtasks for different proportion are presented in Fig. 3 by the line chart. The evaluation scores of our model and JAVE fluctuate less as the data decreased. Even with only 10% of the data for training, our model still performs satisfyingly with 94.53% AP F1 score and 76.89% AVE F1 score. In particular, on the AP task, there is almost no gap between the performances of the models trained on 10% of the data and the full amount of data. This also gives advantages for AVE. But the other generative model JG-AVE is strongly influenced by the amount of

data. On the two subtasks, the f1 score of JG-AVE is lower than our model by over 20%, when 10% training instances are available. Therefore, we can have a conclusion that our model is easy to train and has a stronger learning ability.

## 6 Conclusion

In this paper, we pay attention to the implicit attribute values (IAV) in AVE and propose an effective multi-modal generative-discriminative framework called DEFLATE for attribute value extraction and attribute prediction. Other than the attribute values that are mentioned explicitly (EAV) in the product descriptions, DEFLATE could also acquire those IAV beyond the textual information. We also present a challenging multi-modal dataset for AVE with both EAV and IAV. Extensive experiments demonstrate the superiority of DEFLATE over the previous extractive methods on both AVE and AP tasks, especially for the products with IAV. And our QA-based framework that leverages the relationship between attributes and values on the harder AVE of the two tasks shows an advantage over the jointly generative method JG-AVE, which utilizes it on the easier AP. Besides, the ablation experiments further show the importance of the visual information fusion module and attribute discrimination module in DEFLATE.

## 7 Limitations and Future Work

The limitations of our method are as follows:

(1) Despite the better performances our method DEFLATE achieves on multiple AVE experiments, its mechanism of using attribute as queries needs to construct the same number of sequences as attributes for a target product, which requires more time for training and evaluating when there are particularly many attributes to consider.

(2) The low F1-micro scores of DEFLATE and all other leading methods for AVE on our DESIRE dataset emphasizes the demand of further researches for the information extraction of the e-commerce products with implicit attribute values. And we would explore strategies such as incorporating external knowledge (structured or unstructured) to further enhance the ability of our method on the AVE task in future works.

## References


Min Cao, Sijing Zhou, Honghao Gao, and Youhuizi Li. 2018. A novel hybrid collaborative filtering approach to recommendation using reviews: The product attribute perspective (s). In *SEKE*.

Jason P.C. Chiu and Eric Nichols. 2016. Named entity recognition with bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Jaemin Cho, Jie Lei, Hao Tan, and Mohit Bansal. 2021. Unifying vision-and-language tasks via text generation. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pages 1931–1942. PMLR.

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, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Chris Donahue, Mina Lee, and Percy Liang. 2020. Enabling language models to fill in the blanks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2492–2501, Online. Association for Computational Linguistics.

Chenhe Dong, Yinghui Li, Haifan Gong, Miaoxin Chen, Junxin Li, Ying Shen, and Min Yang. 2023. A survey of natural language generation. *ACM Comput. Surv.*, 55(8):173:1–173:38.

Xin Luna Dong, XIANG He, Andrey Kan, Xian Li, Yan Liang, Jun Ma, Yifan Ethan Xu, Chenwei Zhang, Tong Zhao, Gabriel Blanco Saldana, Saurabh Deshpande, Alexandre Michetti Manduca, Jay Ren, Surender Pal Singh, Fan Xiao, Haw-Shiuan Chang, Giannis Karamanolakis, Yuning Mao, Yaqing Wang, Christos Faloutsos, Andrew McCallum, and Jiawei Han. 2020. Autoknow: self-driving knowledge collection for products of thousands of types. In *KDD 2020*.

Alexey Dosovitskiy, Lucas Beyer, Dirk Weissenborn Alexander Kolesnikov, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020. An image is worth 16x16 words: Transformers for image recognition at scale. *Computer Vision and Pattern Recognition*.

Vishrawas Gopalakrishnan, Suresh Iyengar, Amit Madaan, Rajeev Rastogi, and Srinivasan H. Sengamedu. 2012. Matching product titles using web-based enrichment. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 605–614. ACM.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *ArXiv*, abs/1508.01991.

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*, pages 8489–8502, Online. Association for Computational Linguistics.

Zornitsa Kozareva, Qi Li, Ke Zhai, and Weiwei Guo. 2016. Recognizing salient entities in shopping queries. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 107–111, Berlin, Germany. Association for Computational Linguistics.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, San Diego, California. Association for Computational Linguistics.

Rongmei Lin, Xiang He, Jie Feng, Nasser Zalmout, Yan Liang, Li Xiong, and Xin Dong. 2021. Pam: Understanding product images in cross product category attribute extraction. *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*.

Xuezhe Ma and Eduard Hovy. 2016. End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

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

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. *ArXiv*.

Kalyani Roy, Pawan Goyal, and Manish Pandey. 2021. Attribute value generation from product title using language models. In *Proceedings of the 4th Workshop on e-Commerce and NLP*, pages 13–17, Online. Association for Computational Linguistics.

Kalyani Roy, Tapas Nayak, and Pawan Goyal. 2022. Exploring generative models for joint attribute value extraction from product titles. *Information Retrieval*, arXiv:2208.07130.

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)*, pages 227–234, Dublin, Ireland. Association for Computational Linguistics.

Jianlin Su. 2021. T5 pegasus - zhuiyiai. Technical report, ZhuiyiAI.

Amir Tavanaei, Karim Bouyarmane, Iman Keivanloo, and Ismail Tutar. 2022. Mmt4: Multi modality to text transfer transformer. In *KDD 2022 Workshop on Content Understanding and Generation for E-commerce*.

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

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 *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '20*, pages 47–55.

Bichen Wu, Chenfeng Xu, Xiaoliang Dai, Alvin Wan, Peizhao Zhang, Zhicheng Yan, Masayoshi Tomizuka, Joseph Gonzalez, Kurt Keutzer, and Peter Vajda. 2020. Visual transformers: Token-based image representation and processing for computer vision. *Computer Vision and Pattern Recognition*.

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 Annual Meeting of the Association for Computational Linguistics*, pages 5214–5223, Florence, Italy. 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 *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining (WSDM)*. wsdm.

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*. 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)*, pages 2129–2139, Online. Association for Computational Linguistics.

# A Appendix

## A.1 Implementation Details

The framework of DEFLATE follows T5-pegasus (Su, 2021), which has 275 million parameters. We implement DEFLATE with Pytorch and Huggingface Transformers and train the models on 2*Tesla V100 GPUs. For all models, We set 10 training epochs, the batch size $b = 8$ and the maximum sequence length 60 for tokenizer according to our text data. And an Adamax optimizer with learning rate $l_r = 2e - 3$, $(\beta^1, \beta^2) = (0.9, 0.999)$ and weight decay of 1e-4 is adopted for more efficient training. In all experiments, we train the models with a fixed random seed and make a number of evaluations on the test set using the intermediate checkpoints during training. All of the reported results on DESIRE are selected from the best scores among the 20 evaluation results for each model.

Specifically, for BiLSTM-CRF, since pretraining language models (PLMs) were not popularly used when this method was proposed, we use the word embedding from pretrained BERT (Devlin et al., 2019) model instead of randomized embedding. For M-JAVE, texts and images are preprocessed into vectors using pretrained BERT-base and ResNet (He et al., 2016) models, and then the model is trained and inferred using its publicly available code[2]. In order to allow the above two extractive models to mark parts of implicit attribute values, we select the IAV with at least two consecutive characters that appear in the text and mark these characters as "BIO" tags. And for JG-AVE, we also adopt T5-pegasus (Su, 2021) as the backbone of the generative model and apply only the variant of the word sequence-based paradigm.

[2]https://github.com/jd-aig/JAVE

## ACL 2023 Responsible NLP Checklist

### A For every submission:

☑ A1. Did you describe the limitations of your work?
*7*

☒ A2. Did you discuss any potential risks of your work?
*Our work just provides a method for attribute value extraction from textual and visual information, without offering a pretrained model or a serving system that has the risk of being malicious used by others.*

☑ A3. Do the abstract and introduction summarize the paper's main claims?
*1*

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

### B ☑ Did you use or create scientific artifacts?

*5.1 - 5.3*

☑ B1. Did you cite the creators of artifacts you used?
*5.1 - 5.3*

☑ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
*4 & 5.3*

☑ 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)?
*5.3*

☒ 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?
*The data used in our work does not contain any information about specified individuals.*

☑ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
*4 & 5.3*

☑ 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.
*4*

### C ☑ Did you run computational experiments?

*5*

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

*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?
*Appendix*

☑ 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?
*Appendix*

☑ 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.)?
*Appendix*

**D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?**
*4*

☒ 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.?
*We design an online table where annotators could easily understand and complete their work. So there are no necessary instructions.*

☑ 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)?
*In the supplemental material*

☑ 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?
*4*

☑ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
*4*

☒ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
*The annotators' personal information is not important on our annotation task.*