DiffXtract: Joint Discriminative Product Attribute-Value Extraction

Varun Embar*
University of California, Santa Cruz
vembar@ucsc.edu

Andrey Kan*
University of Adelaide, Australia
andrey.kan@adelaide.edu.au

Bunyamin Sisman
Amazon.com, Inc
bunyamis@amazon.com

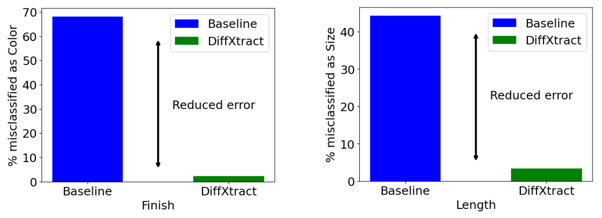
Christos Faloutsos*
Carnegie Mellon University
christos@cs.cmu.edu

Lise Getoor
University of California, Santa Cruz
getoor@ucsc.edu

***Abstract*—Identifying discriminative attributes between product variations, e.g., the same wristwatch models but in different finishes, is crucial for improving e-commerce search engines and recommender systems. Despite the importance of these discriminative attributes, values for such attributes are often not available explicitly and instead are mentioned only in unstructured fields such as product title or product description. In this work, we introduce the novel task of *discriminative attribute extraction* which involves identifying the attributes that distinguish product variations, such as finish, and also, at the same time, extracting the values for these attributes from unstructured text. This task differs from the standard attribute value extraction task that has been well-studied in literature, as in our task we also need to identify the attribute, in addition to finding the value. We propose DiffXtract, a novel end-to-end, deep learning based approach that jointly identifies both the discriminative attribute and extracts its values from the product variations. The proposed approach is trained using a multitask objective and explicitly models the semantic representation of the discriminative attribute and uses it to extract the attribute values. We show that existing product attribute extraction approaches have several drawbacks, both theoretically and empirically. We also introduce a novel dataset based on a corpus of data previously crawled from a large number of e-commerce websites. In our empirical evaluation, we show that DiffXtract outperforms state-of-the-art deep learning-based and dictionary-based attribute extraction approaches by up to 8% F1 score when identifying attributes, and up to 10% F1 score when extracting attribute values.**

44mm Wrist Watch with Stainless Steel Band - Gold
44mm Wrist Watch with Leather Band - Brown

(a) **Product variations: Two wristwatches with different finishes and colors**



(b) Performance for the baseline and **DiffXtract**: The baseline misclassifies finish as color for 70% of the pairs. Similarly, it misclassifies length as size for 40% of the pairs. The proposed **DiffXtract** approach identifies them accurately.

Fig. 1: **Identifying discriminative attributes in variations.**

I. Introduction

How do the two watches in Fig. 1a differ from each other? While they share some canonical attribute values – they have the same manufacturer, brand and model, they have different finishes – `stainless steel` vs. `leather` and color – `gold` vs. `brown`. Such groups of nearly identical but distinct products are called product variations [1].

Identifying attributes that distinguish product variations is crucial for the success of a variety of online platforms. For example, many e-commerce sites provide a way to select product variations in the user interface (UI). This requires the knowledge of discriminative attributes between the variations. Similarly, for conversational search agents, discovering and eliciting a user's choice among several variations is a necessary step for identifying the product the user wishes to purchase. Further, we can also learn user preferences by mining the discriminative attribute values between the products bought by the user and other available variations. This in turn can significantly improve the user's recommendations.

Extracting discriminative attributes from product variations involves several challenges. First, each product is typically associated with a set of structured fields such brand and price and a set of unstructured fields such as title and description. There are usually few structured attributes and these are often incomplete and noisy. Retailers often highlight important attributes of the product by including them in unstructured fields such as title. For example, the information about the finish for the wrist watch bands in Fig. 1a (Stainless steel and leather) might not be present as a structured attribute

*Work done while at Amazon

but are mentioned in the title. While the task of extracting product attribute values from unstructured text has received significant attention [2, 3, 4, 5, 6, 7, 8, 9, 10], using these approaches for discriminative attribute extraction task remains largely unexplored. Fig. 1b shows the performance of a baseline method that extracts discriminative values and then identifies the corresponding attribute (more details in Section IV). We see that this approach incorrectly identifies color as the attribute when the true attribute is finish for about 70% of the cases. Similarly, the baseline identifies size as the attribute instead of length for about 40% of the pairs.

Second, discovering discriminative attributes involves not only extracting the attribute values but also identifying the attribute that differs between products. Different sets of products can have different discriminative attributes. In the case of ``44mm Wrist Watch with Stainless Steel Band - Gold'' and `40mm Wrist Watch with Stainless Steel Band - Gold'' size is the discriminative attribute; however, between the products ``44mm Wrist Watch with Stainless Steel Band - Gold'' and ``44mm Wrist Watch with Stainless Steel Band - Brown'' the discriminative attribute is color. While there is some work on extracting semantic differences between a pair of words [11, 12, 13] and extracting discriminative tokens from product reviews [14], the task of discovering discriminative attributes from product titles has not been studied.

Finally, there are a large number of potential discriminative attributes. Variations in each product category vary along different sets of attributes. Often there is little or no training data for each of the possible discriminative attributes.

In this work, we propose a novel deep learning based approach, **DiffXtract**, to overcome these challenges. **DiffXtract** jointly identifies the discriminative attribute and extracts the attribute values for a given pair of products. The model is trained end-to-end using a multitask objective that combines attribute identification and value extraction tasks. The framework explicitly models the semantic representation of the discriminative attribute and uses attention to capture the relation between the attribute and the product title. This enables the model to scale to a large number of product attributes with little training data. From Fig. 1b, unlike the baseline, we observe that the **DiffXtract** approach identifies finish and length attributes with high accuracy.

The main contributions of our work include:

- **Taxonomy of approaches:** We propose a taxonomy of approaches by extending the state-of-the-art attribute identification and attribute value extraction approaches for the discriminative attribute extraction task.
- **DiffXtract:** We propose a novel end-to-end multitask approach that jointly performs both discriminative attribute identification and value extraction. We establish a theoretical upper bound for a class of approaches called extraction-oriented approaches for the attribute identification task.
- **Effectiveness:** We introduce a novel dataset and empirically show that the proposed **DiffXtract** approach outperforms attribute-oriented and extraction-oriented approaches by up to 8% F1 score when identifying attributes, and up to 10% F1 score when extracting attribute values.

To the best of our knowledge, we are the first to study and propose a solution to the task of discriminative attribute extraction for product variations.

II. Preliminaries

As discussed above, variational attributes play an important role (e.g., in product browsing UI), but their values are often not provided explicitly, and are included in unstructured text fields. In this section, we introduce necessary terms to describe the variational attributes and the formal problem definition. We then provide a discussion of existing approaches for this task.

A. Problem Definition

Product attributes can be broadly divided into two sets of attributes called *base attributes* and *variational attributes* [1]. Canonical attributes such as manufacturer, brand and product line are called **base attributes**. Other product attributes that cater to the users' needs and preferences are called **variational attributes**. Color, size and quantity are some examples of variational attributes. **Product variations** are products that share the same value for all the *base attributes* but have different values for *variational attributes*. For example, the two products in Fig. 1a are variations of each other. The variational attributes finish and color have the values `Stainless steel` and `Gold` for the first product and `Leather` and `Brown` for the second product. However, both the products share the same value, `44mm`, for the variational attribute size. We refer to a variational attribute that has different values among product variations as a *discriminative attribute*. In the example above, finish and color are the discriminative attributes.

Having defined the notion of a discriminative attribute, we now formally define the task of discriminative attribute extraction.

Definition 1. ***Discriminative attribute extraction:*** *Given a pair of product variations with titles (T_1, T_2) the task of discriminative attribute extraction is to output a triple (a_d, v_1, v_2), where a_d is the discriminative attribute for the product pair and (v_1, v_2) are the extracted product attribute values for a_d present in the titles (T_1, T_2).*

For the pair of product variations above, the expected triple is either {`color, Gold, Brown`} or {`finish, Stainless Steel, Leather`}. We assume that the set of all variational attributes $\mathcal{A} = \{a_1, a_2, \cdots, a_m\}$ is given. However, we make an open world assumption for the attribute values. We do not assume any fixed, pre-defined vocabulary of attribute values and products can contain new emerging values that have not been seen before.

We denote the tokens in the product titles T_1, T_2 by the sets $\{t^1_1, t^1_2, \cdots, t^1_m\}, \{t^2_1, t^2_2, \cdots, t^2_n\}$ respectively. The extracted values v_1, v_2 are subsets of the tokens present in T_1, T_2. The set of tokens in the name of the attribute a_i (e.g., color, manufacturer part number) is represented by $\{a^i_1, a^i_2, \cdots, a^i_p\}$.

If one of the product titles contains an attribute value that is missing in the other title, we do not consider this as a discriminative attribute.

B. Discriminative Attribute Extraction Approaches

The discriminative attribute extraction task involves two sub-tasks, identifying the discriminative attribute and extracting the attribute values. Based on the sub-task that is solved first, the approaches can be broadly classified into *extraction-oriented* approaches and *attribute-oriented* approaches.

Extraction-oriented approaches: These approaches extract the attribute values first and then identify the attribute to which these values belong. Since the values of the discriminative attribute needs to be different, these values must contain tokens that are present only in one of the titles. In the example of `''44mm Wrist Watch with Stainless Steel Band - Gold''` and `''40mm Wrist Watch with Stainless Steel Band - Gold''` the tokens `44mm` and `40mm` are present in only one of the titles. Since the attribute values can contain multiple tokens, and some of these tokens can be common to both the titles (e.g., `pack of 2` and `pack of 3`), these approaches start with the unique tokens and grow the values by including adjacent tokens. Once these tokens are identified, extraction-oriented approaches can solve the sub-task of identifying the attribute using state-of-the-art unstructured text to product attribute matching approaches. For example, Kannan et al. [8] use an inverted index that maps values to attribute names. We could also train a multiclass classifier to identify the attribute from the tokens. The main advantage of these approaches is their ability to compare and contrast the product pairs when extracting the values.

Formally, these models can be represented by:

$$(\hat{a_d}, \hat{v_1}, \hat{v_2}) = argmax_{a_d} P(a_d | argmax_{v_1,v_2} P(v_1, v_2 | T_1, T_2)) \tag{1}$$

Attribute-oriented approaches: These approaches identify the attribute first and then extract the values for the identified attribute. These approaches leverage and extend the state-of-the-art product attribute extraction techniques [2, 3, 4, 5, 6, 7, 8, 9, 10]. Since the set of variational attributes is known, these approaches extract the values for each of these attributes and then identify the attribute that has different values. In the example of `''44mm Wrist Watch with Stainless Steel Band- Gold''`, we can extract the values for all variational attributes in $\mathcal{A}$ which includes size, finish, and color. Similarly, we can extract the values for these attributes for the product `''40mm Wrist Watch with Stainless Steel Band- Gold''`. Since the values for the attribute size differ, we return the triplet (size, `44mm`, `40mm`). Formally, these models can be represented by:

$$\begin{aligned}(\hat{a_d}, &\hat{v_1}, \hat{v_2}) \\ &= argmax_{v_1,v_2} P(v_1, v_2 | T_1, T_2, argmax_{a_d} P(a_d | T_1, T_2))\end{aligned} \tag{2}$$

However, both these approaches have several drawbacks. While the attribute-oriented approaches make use of state-of-the-art product attribute extraction techniques, they extract the values for each product independently. They cannot contrast between the product pairs while extracting the values. The extraction-oriented approach, while capable of contrasting between the product pairs, cannot make use of the state-of-the-art product attribute extraction techniques as they require the attribute to be given as an input. Approaches such as Kannan et al. [8], that use an inverted index, find the attribute identification task challenging when provided with previously unseen attribute values.

More importantly, these approaches struggle to identify the correct attribute where the attributes share a large set of values. Attributes such as *color* and *finish*, for example, can take the same set of values such as *red, yellow* and *blue*. These problems are further exacerbated in attributes that take numerical values such as *length*, *width* and *height*. We can quantify the challenge of distinguishing such attributes by calculating the Bayes error rate. The Bayes error rate provides an upper bound on the accuracy that can be achieved by any pattern classifier when class distributions overlap.

Let the attribute $a_i \in \mathcal{A}$ take values from the sets $\mathcal{V}^i = \{v_1^i, v_2^i, \cdots, v_m^i\}$. We denote the probability of attribute a_i taking a value v_k using the conditional distribution $P(v_k|a_i)$. In the case of categorical attributes such as *color* this takes the form of a multinomial distributions Π_i. Let the prior probability of a_i being the true discriminative attribute be denoted by $P(a_i)$.

Theorem 1. *The attribute identification task accuracy for the extraction-oriented approaches has an upper bound given by:*

$$1 - \sum_{v_l, v_m \in \cup_{\mathcal{V}^i}} \big(1 - argmax_i \Pi_i(v_l)\Pi_i(v_m)P(a_i)\big)\Pi_i(v_l)\Pi_i(v_m)$$

where $P(a_i)$ is the probability of a_i being the discriminative attribute and $\Pi_i(v_m)$ is the probability of observing v_m as the value of attribute a_i.

The proof of this theorem is given in the appendix. The accuracy upper bound decreases as the overlapping between values of different attributes increases. The upper bound is also related to the prior probabilities of the attributes. The more similar the prior probabilities of attributes with overlapping values, the lower the upper bound on accuracy.

Multitask approach: To overcome these drawbacks, we propose a multitask approach that jointly performs attribute identification and value extraction. Formally, multitask approaches can be represented by:

$$(\hat{a_d}, \hat{v_1}, \hat{v_2}) = argmax_{a_d,v_1,v_2} P(v_1, v_2, a_d | T_1, T_2) \tag{3}$$

Multitask approaches extract both the values and identify the attribute together. These approaches look at the entire title and model the probability of attribute a_i being the discriminative attribute. The tokens in the title can potentially help these approaches circumvent the problem faced by extraction-based approaches. For example, the position of the values v_l, v_m in the tile T_1, T_2 can provide the context for identifying the attribute. For example, typically, *length* tokens precede *width*

	Attrib. Identification	**Value Extraction**
Attribute-oriented (OpenTag[3], CAM[2], $\cdots$)		✓
Extraction-oriented (Dict [8], $\cdots$)	✓	
DiffXtract (Proposed approach)	✓	✓

TABLE I: Discriminative attribute extraction approaches: The proposed **DiffXtract** approach jointly identifies the attribute and extracts their values

and *height* tokens. Tokens in the title that provide information about the product type can help distinguish between *color* and *finish*.

Table I gives an overview of the different approaches. Attribute-oriented approaches extract the values for a given attribute and extraction-oriented approach identify the attribute given the extracted values. Our proposed multitask approach, **DiffXtract**, jointly performs both tasks and is described in the next section.

III. Multitask approach using **DiffXtract**

Our proposed approach **DiffXtract** uses multitask learning where we train a single end-to-end model to perform multiple tasks. Our model performs all the tasks simultaneously - identifying the discriminative attribute and extracting the values for the identified attribute from each of the product titles. We cast the problem of identifying the discriminative attribute as a classification task, and the task of extracting the attribute values from the two titles as a sequence labeling task. We identify the discriminative attribute and use this as an input for the value extraction tasks.

Our approach follows state-of-the-art neural product attribute extraction techniques and casts the attribute extraction problem as a sequence labeling task. Each token in the title is associated with a label from the set of $\{B, I, O\}$ tags, where B and I denote the beginning and inside tokens of the extracted attribute value respectively, and O denotes the outside of the value tag. Neural attribute value extraction approaches use a bidirectional LSTM (BiLSTM) and conditional random field (CRF) to perform tagging. Unlike dictionary-based approaches that learn from a limited and pre-defined vocabulary of attribute values, sequence labeling approaches can generalize to previously unseen attribute values. Further, we extend the token representation to incorporate information about whether the token is unique to the product or if it is present in both the product titles. This allows the model to compare and contrast the two products when extracting the attribute values.

Fig. 2 shows the architecture of our proposed model. We first review the building blocks of our approach for the classification task, followed by the blocks for the extraction task. We then outline our multitask strategy for the discriminative attribute extraction task.

A. Discriminative Attribute Identification

Word Representation Layer: Neural word embeddings map tokens in a sentence to high dimensional vectors that capture both syntactic and semantic information. We use a pretrained Bidirectional Encoder Representations from Transformers (BERT) [15] to map tokens to vectors. BERT generates contextual word embeddings by taking a sentence as input and maps each token in the sentence to a vector. Since BERT generates contextual embeddings, the embedding for the token *red* in the brand *red bull* is different from the embedding in the color *cherry red*. We add the **[CLS]** and **[SEP]** tokens to the beginning and the end of the title and then generate BERT embeddings for all tokens in both the products.

Bidirectional LSTM Layer: We capture the long-term dependencies between the tokens in the product titles using as Bidirectional LSTM (BiLSTM). Unlike LSTMs, which capture dependencies between a token and its preceding tokens, BiLSTMs capture dependencies in both the directions via backward and forward states. The forward and the backward hidden states are concatenated to form the final output. We use the BERT embeddings of the tokens in the title for each of the products and the same BiLSTM for both titles. The contextual representation of the titles $\mathbf{H^1}, \mathbf{H^2}$ is represented as $\{h^1_{CLS}, h^1_1, h^1_2, \cdots, h^1_m\}$ and $\{h^2_{CLS}, h^2_1, h^2_2, \cdots, h^2_n\}$ and is given by:

$$\mathbf{h^j_i} = [\overrightarrow{h^j_i}; \overleftarrow{h^j_i}] = BiLSTM(\overrightarrow{h^j_{i+1}}, \overleftarrow{h^j_{i-1}}, \mathbf{W_b}) \tag{4}$$

Attribute Classification Layer: We use a softmax layer to predict the discriminative attribute between the titles. We concatenate the hidden representation of **[CLS]** tokens from both the product titles and pass it to the softmax layer. The output is given by:

$$P(a_d = k) = softmax([h^1_{CLS}, h^2_{CLS}].W_h) \tag{5}$$

where W_h is a weight matrix, h^1_{CLS} and h^2_{CLS} are the hidden representations from the BiLSTM layer of **[CLS]** tokens, and $k \in \mathcal{A}$.

Using the training data, we learn the parameters of our model. We use log-likelihood or cross entropy as the loss function as it can handle unbalanced classes in the training set. We denote the log-likelihood of the classifier by l_c.

B. Attribute Value Extraction

Having identified the discriminative attribute, we next extract the values for this attribute from each of the titles. Similar to the discriminative attribute identification task, we first encode the tokens of the title into high dimensional vectors using BERT and pass the embeddings to a BiLSTM to generate the vector representation for the tokens. Unlike the classification task, we need to generate tags for each token in the title. Hence, we use all hidden states of BiLSTM from both the titles, i.e., $\mathbf{H^1}, \mathbf{H^2}$.

Attribute representation: Similar to the tokens in the title, we generate contextual representations for all the attributes in $\mathcal{A}$. We first map the tokens present in the attribute names to high dimensional vectors using BERT and pass them to another BiLSTM and use the hidden representation of the last token as the representation of the attribute. We represent the contextual vectors for each of the attribute by h^a where $a \in \mathcal{A}$.

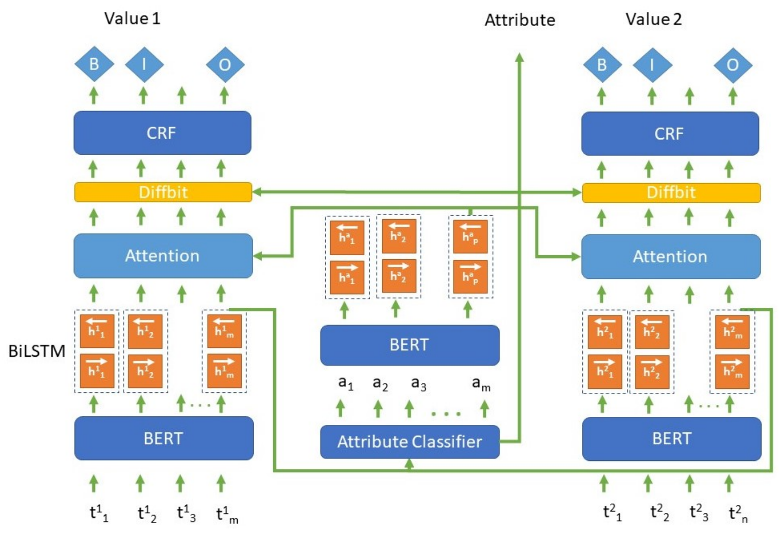


Fig. 2: **DiffXtract: Discriminative Attribute Extraction Model**

From the output of the attribute identification task, we select the attribute with the highest probability as the discriminative attribute and use its representation for the value extraction task. The discriminative attribute is represented as h^{a_d} and is given by:

$$h^{a_d} = h^{\hat{a}} \text{ where } \hat{a} = argmax_a P(a_d = a) \tag{6}$$

Attention Layer: While generating the final set of tags, the CRF considers all tokens to be equally important. However, this is not true as some tokens are more important when extracting the attribute values. In the Neural Machine Translation literature, an attention mechanism was first used with great success by Bahdanau et al. [16]. An attention mechanism enables the model to *attend* to different parts of the input while generating output tags.

While generating the output tags for the tokens in the title, the attribute-based attention mechanism enables us to attend to different tokens in the title while extracting values for different attributes. This allows us to extract values for all the attributes in $\mathcal{A}$ without training a separate model for each of the attributes.

We compute the similarity between the attribute and the product title token representations to obtain attention weights $A = \{\alpha_1, \alpha_2 \cdots, \alpha_m\}$. We use cosine similarity between the attribute representation and the token representation, i.e, $\alpha_i = cosine(h^{a_d}, h_i^t)$.

The attribute-weighted title representation is given by $\mathbf{C^i} = A \odot \mathbf{H}^i$, where $\odot$ represents element-wise multiplication. C^i represents the weighted sum of words in the title T^i with respect to the attribute a_d.

Diffbit: Tokens that are present in one title but not the other are likely to be part of the discriminative attribute value. To provide this signal to the final layer, we compute a diffbit for each token in the title. This bit is 1 if the token is not present in the other title and set to 0 otherwise. We denote this by d^t. We append the diffbits to the token representations.

Output layer: The final output layer generates the BIO tags that are used to extract the attribute values. We use conditional random fields (CRF) [17] for this task as they capture dependencies in the output labels. For example, if the tag for a token is O, we know that the probability of tag I for the next token is 0. We concatenate the title representations for the BiLSTM $\mathbf{H^t}$, the attribute comprehension title $\mathbf{C^t}$ and the diffbit d^t to obtain the feature matrix $\mathbf{M^t}$:

$$\mathbf{M^t} = [\mathbf{H^t}, \mathbf{C^t}, d^t] \tag{7}$$

This feature matrix is used by the CRF layer to generate the list of tags for each token in the title. The joint probability distribution of tags y is given by:

$$P(y_i|T;\psi) \propto \prod_{i=1}^{m} exp(\sum_{k=1}^{K} \psi_k f_k(y_{i-1}, y_i, \mathbf{M^t})) \tag{8}$$

where ψ_k corresponds to the feature weight, f_k is the feature function, and K is the number of features. The final output is the best label sequence y^* with the highest conditional probability, i.e.:

$$y^* = argmax_y P(y|T;\psi) \tag{9}$$

We learn the parameters of the model using the maximum conditional log-likelihood estimate. The maximum conditional likelihood is given by:

$$l_v(\psi) = \sum_{i=1}^{N} log\ P(y_i|T_i;\psi) \tag{10}$$

where N denotes the number of training samples. We denote the log-likelihood for the first product as l_{v^1} and for the second product as l_{v^2}.

C. Multitask Learning

Having described the building blocks of the classification and extraction model, we now describe the full model that

Attribute	Train	Validation	Test
Color	15357	3225	5316
Manufacturer part number	9203	1009	2071
Model	6062	824	970
Finish	5766	542	630
Size	1725	201	260
Length	1047	88	123
Width	975	92	128
Depth	882	50	72
Style	518	33	50
Material	254	23	42
Type	232	80	124
Dimensions	116	3	22
Height	107	6	17
Product type	91	12	21
UPC	61	3	6
Condition	21	16	56
Weight	15	3	1
Features	9	2	0
ASIN	4	0	0
Transmission	2	1	1

TABLE II: Attribute distribution: The table show the distribution of discriminative attributes across train, validation and test splits. Here, ASIN refers to Amazon Standard Identification Number and UPC refers to Universal Product Code.

is trained end-to-end. Fig. 2 shows the architecture of our proposed model. We jointly train the model for discriminative attribute identification and attribute value extraction. We do this by combining the likelihood functions l_c, l_v^1 and l_v^2. The multitask likelihood objective is given by:

$$l = l_c + \lambda(l_v^1 + l_v^2) \tag{11}$$

where $\lambda \in \mathcal{R}$ is a hyperparameter that trades-off between the attribute identification task and attribute value extraction task.

At prediction time, we first identify the discriminating attribute using the hidden representation of the **[CLS]** tokens from both the products. Having identified the attribute by contrasting between the two titles, we extract the attribute value. The semantic representation of the identified attribute in the attention layer and the diffbit token enables the approach to extract the value for attributes with few training examples.

IV. Experimental Evaluation

In this section, we perform experimental evaluation to answer the following questions

- **Q1:** How does the proposed **DiffXtract** method compare to other techniques for the task of discriminative attribute extraction?
- **Q2:** What attributes are harder to identify and extract?
- **Q3:** How does the proposed **DiffXtract** method perform on the sub-task of attribute extraction?
- **Q4:** How sensitive is the hyperparameter λ that trades-off between the two sub-tasks?

A. Data

We performed our experimental evaluation using the Multi-modal Attribute Extraction dataset [18]. The dataset contains over 2.2 million products collected from several e-commerce sites. The dataset includes products from various categories such as electronics, jewelry, clothing and vehicles. Along with the product title, the dataset provides an open-schema table of attribute-value pairs. There are about 7.6 million attribute-value pairs that span 2100 attributes.

Within an e-commerce platform, the information about which products are variations is often provided by sellers. In our evaluation, we identified product variations with a previously used approach [1]. We do this by first splitting the products into train, validation and test splits in the ratio $0.8 : 0.05 : 0.15$. For each of the splits, we blocked the products using the tokens in the titles, and extracted pairs where the title Jaccard similarity is ≥ 0.7 and have the same value for brand. From these products, we identified product attributes that have a frequency greater than 5000. The list of identified attributes is given in Table II. For these attributes, we identified pairs where each product in the pair contains the same attribute with different values among the attribute-value pairs associated with the products. For example, the products `''clear kaleidoscope static cling window film 35′ wide x 75 ft''` and `''clear kaleidoscope static cling window film 35′ wide x 82 ft''` were associated with the attribute `length` with values `75 ft` and `82 ft`. Their common attribute with different values is the ground truth for the discriminating attributes. Further, we removed pairs where the values of the discriminating attribute values were not present in the title. There were 42,447 pairs in the train split, 6215 pairs in the validation split and 9910 pairs in the test split. The attribute distribution across splits is given in Table II.

We compute the precision, recall and F1 scores for the attribute identification task and the value extraction tasks. For the extraction task, we use an *Exact Match* criteria were the model gets credit only when the full sequence of extracted values are correct. As the attribute distribution is skewed, we compute the weighted macro-average where metrics are computed for each attribute and are weighted by the attribute's support.

B. Approaches

We analyse the performance of extraction-oriented and attribute-oriented baselines, and compare them with the proposed **DiffXtract** model.

Dict: For the extraction-oriented approach, we evaluated an inverted index approach (**Dict**), similar to Kannan et al. [8], where we generate a dictionary containing the values that an attribute can take. We use the train and validation splits to generate this dictionary. At test time, for each of the pairs, we first identify discriminative tokens present in one of the products but not the other. For each of the products, next we identify attributes whose values are present in the title and also contain a discriminative token. We then identify common attributes that were extracted from both the products and sort them based on the sum of frequency of the attribute values. We return the top attribute and its values as the discriminative attribute.

Opentag: For the attribute-oriented approach, we compare our approach with **Opentag** [3], a recently introduced neural sequence tagging model. This approach extends the BiLSTM-CRF architecture by adding a self-attention mechanism to highlight important information before the CRF layer. To ensure fair comparison with the other approaches, we used BERT word embeddings instead of the GloVe embeddings[19] proposed in the model. We use the best performing joint multi-attribute extraction model that extracts values for all possible attributes. We extract attribute-value pairs for both the titles and select the attribute with different values as the discriminative attribute value triplet. We use the training set to train the model and use the validation set to perform early stopping. We evaluate the validation set after every epoch and stop the training if the F1 score for the attribute identification task or the value extraction task decreases for three consecutive epochs.

Contextual attribute extraction model : We also extend the state-of-the-art contextual attribute extraction model (**CAM**) [2]. **CAM** is a product attribute extraction approach that takes as input a product title and an attribute and extracts the attribute value from the product title. We extend this model by first extracting the value of all possible discriminative attributes from the title. We then identify attributes where the extracted values are different and rank the attributes based on the likelihood scores. We return the top ranked attribute along with the values as the discriminative attribute value triplet. Since this approach extracts values for each product independently, we trained the model by passing each product title in the pair and the attribute as a datapoint. We use the training set to train the model and use the validation set to perform early stopping. We evaluate the validation set after every epoch and stop the training if the F1 score for the attribute identification task or the value extraction task decreases for three consecutive epochs.

DiffXtract: This is the proposed multitask approach that jointly identifies the discriminative attribute and extracts values for that attribute. We select the attribute with the highest probability as the discriminative attribute along with the extracted values. We set parameter λ, that trades-off between the attribute identification task and attribute value extraction task, to 5. We use the training set to train the model and use the validation set to perform early stopping. Similar to the above approaches we perform early stopping using the validation split.

For **DiffXtract**, **CAM** and **Opentag**, we set the batch size to 256, and a learning rate of 0.01. We set the hidden dimension to 100 for the BiLSTM layer. During training, the maximum number of epochs was set to 35, although all approaches converged before that. We trained our models on a machine that had a 250GB RAM, Intel Xeon 3.20GHz CPU and a NVIDIA Quadro RTX 6000 GPU.

C. *Q1:* *Discriminative Attribute Extraction Performance*

The precision, recall and F1 scores for the attribute identification and value extraction tasks are shown in Table III.

	Attribute identification			Value extraction		
	Prec	Recall	F1	Prec	Recall	F1
Dict	0.855	0.632	0.646	0.721	0.558	0.629
Opentag	0.869	0.842	0.835	0.768	0.676	0.719
CAM	0.346	0.350	0.323	0.820	0.526	0.641
DiffXtract (No d^t)	0.853	0.875	0.858	**0.873**	0.712	0.784
DiffXtract	**0.908**	**0.912**	**0.908**	0.837	**0.751**	**0.792**

TABLE III: **Performance metrics:** The table shows the precision, recall and F1 scores for the attribute identification and value extraction tasks. We observe that the **DiffXtract** approach outperforms all other approaches.

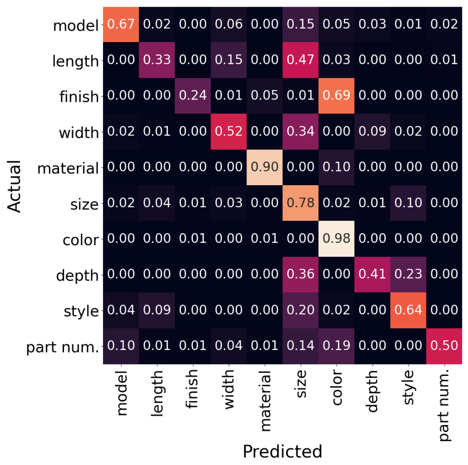
Attribute identification task: We observe that **DiffXtract** outperforms all other approaches by more than 8% on the F1 score. **CAM** performs poorly compared to all other approaches on this task. This is because the model is not trained to explicitly perform this task, and likelihood scores are not always informative with respect to attribute identification. Further, **Opentag** and **CAM** models identify attributes from each product independently and hence are unable to contrast between the pairs. **Opentag** has a separate tag for each attribute and hence is able to identify the attribute better than **CAM**. **Dict** contrasts between product pairs by identifying attribute values present in one pair but not the other. As a result, **Dict** performs better on the attribute identification task compared to **CAM**. However, **Dict** does not take into account the other tokens in a title when identifying the attribute. As a result, it performs poorly in the attribute identification task compared to **DiffXtract**. Further, we observe that adding the diffbit (d^t) results in a performance improvement of 5% on the F1 Score.

The upper bound on the precision for the Bayes classifier mentioned in Theorem 1 on this dataset is 0.93. We note that the precision for **Dict** (0.64) is lower than the best possible value. We also observe that the precision for **DiffXtract** (0.908) is much close to this, although this approach makes use of the context in the title and can have precision higher than the Bayes classifier.

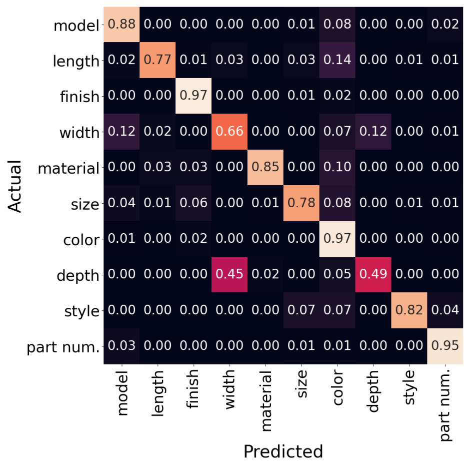
Value extraction task: We observe that **DiffXtract** outperforms all other approaches by more than 10% on the F1 score. **Dict** performs poorly compared to all other approaches on this task. The **Dict** approach makes a closed-world assumption and cannot discover new attributes that are not present in the training data. As a result, it suffers from low recall which hurts its performance. **Opentag** does not make a closed world assumption and hence has a much higher recall. While **CAM** also does not make the closed-world assumption, the incorrect attribute identification, which is used to attend during value extraction, hurts the performance of value extraction. As in the attribute identification task, we observe that adding the diffbit improves performance.

D. *Q2:* *Analysis of Attribute Identification Task*

To answer **Q2**, we analyze metrics for the top 10 attributes in $\mathcal{A}$ by computing the confusion matrix. Fig. 3 shows the confusion matrix for the **Dict** and **DiffXtract** approaches. The



(a) Confusion matrix for **Dict**



(b) Confusion matrix for **DiffXtract**

Fig. 3: **Confusion matrix: Dict** is unable to distinguish between attributes with overlapping values. **DiffXtract** correctly identify the attributes even when they have similar values.

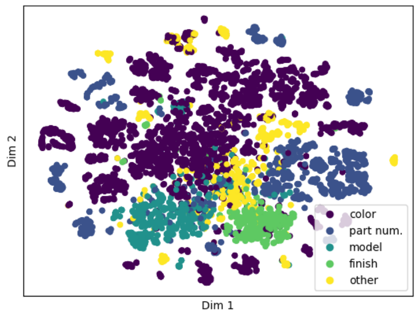


Fig. 4: **t-SNE plot:** Plot shows the hidden states of **[CLS]** tokens from both the products. Attributes forms distinct clusters.

diagonal elements show the fraction of test pairs for each attribute that were correctly classified and the off-diagonal elements show the incorrectly classified pairs. We see that both approaches have high diagonal elements for some attributes suggesting that they perform the attribute identification task reasonably well for these attributes. However, for attributes that take similar values, we observe **Dict** is unable to distinguish between them. For example, the attribute *finish* is confused for *color* in about 70% of the pairs. Similarly, the attribute for *length*, *depth* and *width* is confused for *size*. This is expected as these attributes are numerical in nature. The **DiffXtract** model captures the token positions and context information from the other tokens in the titles. As a result, **DiffXtract** model correctly identifies the attributes even when they have similar values.

To confirm this hypothesis, we plot the vector used by **DiffXtract** to identify the attribute. This contains the hidden states of **[CLS]** tokens from both the products. We project this to a two-dimensional space using t-SNE [20]. The plot for the top four attributes is shown in Fig. 4. We observe that variations that differ by color and those that vary by finish form two distinct clusters. The model makes use of other tokens present in the title to distinguish them. Similarly, we observe variations that differ by part number and model also form different clusters.

	Precision	Recall	F1
DiffXtract (No d^t)	0.857	0.813	0.843
DiffXtract	**0.889**	**0.839**	**0.863**

TABLE IV: **Attribute extraction Metrics:** Providing the ground-truth improves the attribute extraction task F1 performance by 8%.

E. Q3: Product Attribute Extraction

We perform ablation experiments and evaluate the sub-task of product attribute extraction. We provided the ground-truth attribute and evaluate the value extraction performance. The extraction metrics for the the proposed **DiffXtract** model are given in Table IV. Comparing with the metrics in Table III, we observe the F1 metric for the **DiffXtract** models improves by 8%. We observe this for both the full model and the model without the diffbit (d^t). This shows that identifying the correct attribute has a significant impact on the value extraction task. The **DiffXtract** model has higher F1 score which suggests the diffbit helps in value extraction.

F. Q4: Sensitivity to λ

To answer Q4, we train the **DiffXtract** model with different values for λ and plot the precision, recall and F1 metrics for the two sub-tasks. The different metrics for varying values of λ are shown in Fig. 5. The model is robust to different values of λ. Varying the values led to slightly different epochs before the model converged.

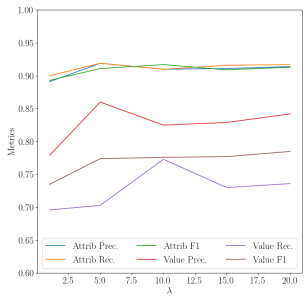


Fig. 5: **Sensitivity to λ:** The model is robust to λ.

V. Related Work

In this section we first review the related work in product attribute extraction followed by discriminative attribute extraction.

A. Product attribute extraction

The task of extracting entity attribute values from unstructured text has received significant attention [2, 3, 4, 5, 6, 7, 8, 9, 10, 21]. These approaches can be broadly classified into closed-world approaches which assume a predefined set of attribute values [4, 6, 7], and open-world approaches which do not make such assumptions [2, 3, 22]. Rule-based and linguistic approaches such as Chiticariu et al. [23], Nadeau and Sekine [24], Mikheev et al. [25] leverage the syntactic structure of the text to extract the attributes. CRF-based systems such as Putthividhya and Hu [7] make use of a seed dictionary to bootstrap the models. Recently, neural network-based models that combine LSTMs and CRF have been proposed [2, 3, 22, 26]. Wang et al. [27] propose a multitask question answering model that casts the attribute extraction task as an answer span identification task. Zhu et al. [21] propose a multimodal approach that uses both the product description and the image to extract attributes. These approaches extract the value for a given attribute from a product title. Our task involves contrasting between two product variations and extracting both the discriminative attribute and its values.

B. Discriminative attribute extraction

McRae et al. [28] was one of the early works that studied the task of identifying important features of living and non-living entities. They introduced the notion of semantic feature production norms or McRae norms to test theories of semantic representation and computation. They collected a list of features that people think are important for set of 541 living and non-living concepts. Sommerauer and Fokkens [11] proposed an approach for investigating the nature of semantic information captured by word embeddings. Stepanjans and Freitas [12] proposed a model to explicitly detect and explain discriminative attributes from word embeddings. Krebs et al. [13] first proposed the task of semantic difference detection where the goal is to predict whether a given word could discriminate between two words. They model the semantic difference as a ternary relation between two concepts such as *apple* and *banana* and a discriminative attribute such as *red* that characterizes the first concept but not the other. Kim and Kang [14] proposed a Latent Dirichlet Allocation-based approach to extract discriminative attributes from product reviews. Building on this literature, we propose the task of discriminative attribute extraction for product variations where when given a pair of product titles, the goal is to identify the discriminative attribute and extract the values from the titles.

VI. Conclusion and Future Work

In this work, we proposed the novel task of discriminative attribute extraction that is crucial for product search engines and voice-based shopping assistants. We showed that approaches for this task can be classified into three groups, extraction-oriented approaches, attribute-oriented approaches and multitask approaches. We then proposed a novel multitask approach that jointly identifies and extracts the attributes. For extraction-oriented approaches, we proved a theoretical upper bound for the attribute identification task. Empirical results on a real-world product dataset show that the proposed multitask approach outperforms all other approaches.

This work suggests other interesting future directions. The proposed approach can be extended to learn the preferences of a user by identifying the discriminative attribute values between the products bought by the user and other available variations. Generating such user profiles and using them to improve the recommendations is an interesting direction. Another direction for future work is to use the proposed approach to build a conversational search agent that can elicit user's preferences and identify the correct variation the user would be likely to purchase.

VII. Acknowledgements

This work was partially supported by the National Science Foundation grants CCF-1740850, CCF-2023495 and IIS-1703331, AFRL, IFDS DMS-2023495, and the Defense Advanced Research Projects Agency and an Amazon Research Award.

References

[1] V. Embar, B. Sisman, H. Wei, X. L. Dong, C. Faloutsos, and L. Getoor, "Contrastive entity linkage: Mining variational attributes from large catalogs for entity linkage," in *AKBC*, 2020.

[2] H. Xu, W. Wang, X. Mao, X. Jiang, and M. Lan, "Scaling up open tagging from tens to thousands: Comprehension empowered attribute value extraction from product title," in *ACL*, 2019.

[3] G. Zheng, S. Mukherjee, X. L. Dong, and F. Li, "Opentag: Open attribute value extraction from product profiles," in *KDD*, 2018.

[4] R. Ghani, K. Probst, Y. Liu, M. Krema, and A. Fano, "Text mining for product attribute extraction," *KDD Explorations Newsletter*, 2006.

[5] P. Petrovski and C. Bizer, "Extracting attribute-value pairs from product specifications on the web," in *WI*, 2017.

[6] X. Ling and D. S. Weld, "Fine-grained entity recognition." in *AAAI*, 2012.

[7] D. P. Putthividhya and J. Hu, "Bootstrapped named entity recognition for product attribute extraction," in *EMNLP*, 2011.

[8] A. Kannan, I. E. Givoni, R. Agrawal, and A. Fuxman, "Matching unstructured product offers to structured product specifications," in *KDD*, 2011.

[9] H. Köpcke, A. Thor, S. Thomas, and E. Rahm, "Tailoring entity resolution for matching product offers," in *EDBT*, 2012.

[10] P. Ristoski, P. Petrovski, P. Mika, and H. Paulheim, "A machine learning approach for product matching and categorization," *SWJ*, vol. 9, pp. 1–22, 2017.

[11] P. Sommerauer and A. Fokkens, "Firearms and tigers are dangerous, kitchen knives and zebras are not: Testing whether word embeddings can tell," in *EMNLP Workshop on BlackboxNLP*, 2018.

[12] A. Stepanjans and A. Freitas, "Identifying and explaining discriminative attributes," in *EMNLP*, 2019.

[13] A. Krebs, A. Lenci, and D. Paperno, "Semeval-2018 task 10: Capturing discriminative attributes," in *SEMEVAL*, 2018.

[14] S. G. Kim and J. Kang, "Analyzing the discriminative attributes of products using text mining focused on cosmetic reviews," *Information Processing & Management*, vol. 54, no. 6, pp. 938–957, 2018.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *NAACL*, 2019.

[16] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *ICLR*, 2015.

[17] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," *ICML*, 2001.

[18] R. L. Logan IV, S. Humeau, and S. Singh, "Multimodal attribute extraction," *AKBC*, 2017.

[19] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *EMNLP*, 2014.

[20] L. Van der Maaten and G. Hinton, "Visualizing data using t-sne." *Journal of machine learning research*, vol. 9, no. 11, 2008.

[21] T. Zhu, Y. Wang, H. Li, Y. Wu, X. He, and B. Zhou, "Multimodal joint attribute prediction and value extraction for e-commerce product," *EMNLP*, 2020.

[22] G. Karamanolakis, J. Ma, and X. L. Dong, "Txtract: Taxonomy-aware knowledge extraction for thousands of product categories," *KDD*, 2020.

[23] L. Chiticariu, R. Krishnamurthy, Y. Li, F. Reiss, and S. Vaithyanathan, "Domain adaptation of rule-based annotators for named-entity recognition tasks," in *EMNLP*, 2010.

[24] D. Nadeau and S. Sekine, "A survey of named entity recognition and classification," *Lingvisticae Investigationes*, vol. 30, pp. 3–26, 2007.

[25] A. Mikheev, M. Moens, and C. Grover, "Named entity recognition without gazetteers," in *EACL*, 1999.

[26] Z. Kozareva, Q. Li, K. Zhai, and W. Guo, "Recognizing salient entities in shopping queries," in *ACL*, 2016.

[27] Q. Wang, L. Yang, B. Kanagal, S. Sanghai, D. Sivakumar, B. Shu, Z. Yu, and J. Elsas, "Learning to extract attribute value from product via question answering: A multi-task approach," in *KDD*, 2020.

[28] K. McRae, G. S. Cree, M. S. Seidenberg, and C. McNorgan, "Semantic feature production norms for a large set of living and nonliving things," *Behavior research methods*, vol. 37, no. 4, pp. 547–559, 2005.

Appendix

Theorem. *The attribute identification task accuracy for the extraction-oriented approaches has an upper bound given by* $1 - \sum_{v_l, v_m \in \cup_{\mathcal{V}i}} [1 - argmax_i \Pi_i(v_l)\Pi_i(v_m)P(a_i)]\Pi_i(v_l)\Pi_i(v_m)]$ *where* $P(a_i)$ *is the probability of* a_i *being the discriminative attribute and* $\Pi_i(v_m)$ *is the probability of observing* v_m *as the value of attribute* a_i.

Proof. Extraction-oriented approaches first extract the values and then identify the attributes, i.e:

$$(\hat{a_d}, \hat{v_1}, \hat{v_2}) = argmax_{a_d} P(a_d | argmax_{v_1,v_2} P(v_1, v_2 | T_1, T_2))$$

The attribute-identification task for these approaches assuming correct extraction is given by $argmax_{a_d} P(a_d | v_1, v_2)$.

From Bayes rule, the posterior probability of observing the triplet (a_i, v_l, v_m) denoted by $P(a_i|v_l, v_m)$ is given by:

$$P(a_i|v_l, v_m) = \frac{P(v_l, v_m|a_i)}{\sum_a P(v_l, v_m|a_a)P(a_a)}$$

Since the values v_l, v_m*are independent conditioned on* a_i

$$= \frac{P(v_l|a_i)P(v_m|a_i)P(a_i)}{\sum_a P(v_l|a_a)P(v_m|a_a)P(a_a)}$$
$$= \frac{\Pi_i(v_l)\Pi_i(v_m)P(a_i)}{\sum_a \Pi_a(v_l)\Pi_a(v_m)P(a_a)}$$

The optimal Bayes classifier, assign the attribute with the highest posterior probability as the discriminative attribute i.e.:

$$\hat{a_d} = argmax_{a_i} \frac{\Pi_i(v_l)\Pi_i(v_m)P(a_i)}{\sum_a \Pi_a(v_l)\Pi_a(v_m)P(a_a)}$$
$$= argmax_i \Pi_i(v_l)\Pi_i(v_m)P(a_i)$$

The classifier misclassifies when the true discriminative attribute does not have the highest posterior probability. This is called the Bayes error rate and is given by:

$$E_{bayes} = \sum_{v_l, v_m \in \cup_{\mathcal{V}i}} [1 - argmax_i \Pi_i(v_l)\Pi_i(v_m)P(a_i)]\Pi_i(v_l)\Pi_i(v_m)$$

Thus the accuracy of the classifier is upper bounded by $1 - E_{bayes}$, i.e:

$$1 - \sum_{v_l, v_m \in \cup_{\mathcal{V}i}} [1 - argmax_i \Pi_i(v_l)\Pi_i(v_m)P(a_i)]\Pi_i(v_l)\Pi_i(v_m)]$$

□