Extracting Attribute-Value Pairs from Product Specifications on the Web

Petar Petrovski
Data and Science Group,
University of Mannheim
B6, 26
Mannheim, Germany 68159
petar@informatik.uni-mannheim.de

Christian Bizer
Data and Science Group,
University of Mannheim
B6, 26
Mannheim, Germany 68159
chris@informatik.uni-mannheim.de

ABSTRACT

Comparison shopping portals integrate product offers from large numbers of e-shops in order to support consumers in their buying decisions. Product offers often consist of a *title* and a free-text *product description*, both describing product attributes that are considered relevant by the specific vendor. In addition, product offers might contain structured or semi-structured *product specifications* in the form of HTML tables and HTML lists. As product specifications often cover more product attributes than free-text descriptions, being able to extract attribute-value pairs from these specifications is a critical prerequisite for achieving good results in tasks such as product matching, product categorisation, faceted product search, and product recommendation.

In this paper, we present an approach for extracting attribute-value pairs from product specifications on the Web. We use supervised learning to classify the HTML tables and HTML lists within a web page as product specification or not. In order to extract attribute-value pairs from the HTML fragments identified by the specification detector, we again use supervised learning to classify columns as attribute column or value column. Compared to DEXTER, the current state-of-the-art approach for extracting attribute-value pairs from product specifications, we introduce several new features for specification detection and support the extraction of attribute-value pairs from specifications having more than two columns. This allows us to improve the F-score up to 10% for extracting attribute-value pairs from tables and up to 3% for lists. In addition, we report the results of using duplicate-based schema matching to align the product attribute schemata of 32 different e-shops. This experiment confirms the suitability of duplicate-based schema matching for product data integration.

CCS CONCEPTS

• **Information systems → Extraction, transformation and loading**;

(a) Example of a free-text product title

Item specifics

Condition:	New: A brand-new, unused, unopened, undamaged item in its original packaging (where packaging is … Read more	Brand:	Samsung
UPC:	610214632623	Processor:	Quad Core
MPN:	SGH-M919	Color:	Black
Model:	Samsung Galaxy S4	Network:	T-Mobile
Contract:	Without Contract	Storage Capacity:	16GB
Features:	3G Data Capable, 4G Data Capable, Bluetooth Enabled, Internet Browser, Music Player, Speakerphone, Touchscreen, Wi-Fi Capable, Voice-Activated Dialing	Screen Size:	5"
Operating System:	Android	Style:	Smartphone
Camera:	13.0MP	Camera Resolution:	13.0MP
Carrier:	T-Mobile	Lock Status:	Network Locked
Cellular Band:	WCDMA (UMTS) / GSM 850/900/1800/1900		

(b) Example of a product specification table

Figure 1: Product attributes within a free-text product title and product attributes given by a specification table

KEYWORDS

Product Data; Feature Extraction; Schema Matching; Web Tables

ACM Reference format:
Petar Petrovski and Christian Bizer. 2017. Extracting Attribute-Value Pairs from Product Specifications on the Web. In *Proceedings of Web Intelligence (WI'17), Leipzig, Germany, August 2017,* 8 pages.
DOI: 10.1145/3106426.3106449

1 INTRODUCTION

The Web has made it easier for organisations to reach out to their customers, eliminating barriers of geographical location[1], and leading to a steady growth of e-commerce sales[2] . Besides e-shops run by individual vendors, comparison shopping portals which aggregate offers from multiple vendors play a central role in e-commerce.

The central challenge for many tasks within the domain of e-commerce, including product matching, product categorisation, faceted product search, and product recommendation, is extracting attribute-value pairs with high precision from unstructured

[1]Digital buyer penetration worldwide from 2011 to 2018 - http://www.emarketer.com/public_media/docs/eMarketer_eTailWest2016_Worldwide_ECommerce_Report.pdf
[2]Retail e-commerce sales worldwide from 2014 to 2019 - https://www.emarketer.com/Article/Worldwide-Retail-Ecommerce-Sales-Will-Reach-1915-trillion-This-Year/1014369

Web Intelligence (WI'17), Leipzig, Germany

DOI: 10.1145/3106426.3106449

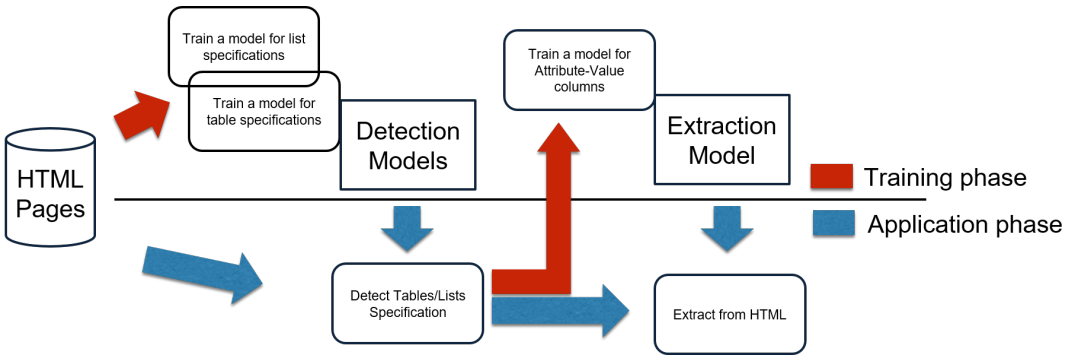

Figure 2: Overall process of specification detection and attribute-value pair extraction

product descriptions or semi-structured product specifications. The extraction of detailed product features from the HTML pages is challenging, as a single feature may appear in various surface forms in headlines, the product name, free-text product descriptions, as well as structured or semi-structured product specifications. Moreover, different vendors use different schemata to describe products from the same product category, in which case to successfully determine matching products schema alignment is needed. Product feature extraction is difficult as most e-shops publish heterogeneous product descriptions having different levels of detail [15].

In [18], we proposed an approach for matching product offers from HTML pages that contain semantic annotations (i.e Microdata markup). We observed that the product titles and product descriptions on these pages often do not include enough attributes by which the specific products can be identified. For example, Figure 1a shows a product title from which only five attributes can be extracted. These attributes are not enough to identify the specific product: Namely, the Galaxy S4 can have a different storage capacity (32 GB or 64 GB) and be equipped with a different chip set (Cortex-A57 or Cortex-A53). This makes Galaxy S4 phones having different storage capacities or different chip sets ultimately different products. In contrast to product tiles and product descriptions which often contain only are relatively small number of attributes, product specifications, such as the HTML table shown in Figure 1b, often contain a much larger number of very detailed product attributes, e.g. 20 attributes in the example.

In order to overcome the challenge of the low number of attributes covered by product descriptions, we present in this paper a two step approach for extracting attribute-value pairs from technical specifications rather than from free-text product descriptions. In addition, we demonstrate a matching method for aligning the product attribute schemata used by different vendors. Thus the contributions of this paper are the following:

Specification Detection: As previous work, we also use supervised learning to classify the HTML tables and HTML lists within a web page as product specifications or not. We build on the state of the art method for this task (DEXTER [21]) and introduce a set of additionall features which allow us to improve the F-score for the detection task by up to 5%.

Specification Extraction: In order to extract attribute-value pairs from the HTML fragments identified by the specification detector, we again use supervised learning to classify columns as an attribute or value column. In contrast to the state-of-the-art system [21], we also support the extraction of attribute-value pairs from specifications having more than two columns.

Schema Alignment: We employ a duplicate-based schema matching approach to align the product attribute schemata of 32 different e-shops. This experiment confirms the suitability of duplicate-based methods for matching product attribute schemata.

The rest of this paper is structured as follows. Section 2 defines the tasks of product feature extraction and product schema alignment. We proceed by giving an overview of the related work in Section 3. Section 4 describes our methods for product feature extraction and product schema alignment. Subsequently, Section 5 presents evaluation of the methods using a Web Data Commons gold standard for product matching and product feature extraction. We conclude with a summary and an outlook on future work.

2 PROBLEM STATEMENT

Definition 1. Product Feature Extraction from Technical Specifications: We have a set of HTML pages W extracted from the Web. Every record $s \in W$ consists of a *product title* and a *product description* as unstructured textual fields, as well as a technical specification embedded in a HTML table or list. Our objective is extract attribute-value pairs from the embedded technical specifications of the records in W. More precisely, we first build a model that can detect the specifications in the records in W. Next, we build a feature extraction model able to extract attribute-value pairs from the technical specifications in W. After the feature extraction model is applied, each product specification $s \in W$ is represented as a list of attributes-values pairs $A_p = \left\{a_1^w, a_2^w, ..., a_n^w\right\}$, where the attributes are numerical or categorical.

Definition 2. Product Schema Alignment: We have a centralised product catalog C of structured product specifications. Every record $r \in C$ consists of a set of attribute-value pairs $R_p = \left\{a_1^c, a_2^c, ..., a_n^c\right\}$, where the attributes are numeric, categorical, or free-text. Our goal is to use the product set C as a background

knowledge to infer schema alignment rules to align the schemas for the extracted attribute-value pair set A_p from W. More precisely, let A^c be an attribute from the catalog schema. Let A^w be an attribute from the schema of record from W (i.e., A^w appears in specifications for a given vendor in W). We say that (A^c, A^w) is an attribute correspondence from $A^c \rightarrow A^w$ if A^c and A^w have the same meaning in their values given a similarity function $s(v_{A^c}, v_{A^w})$.

3 RELATED WORK

This section gives an overview of the existing research on product feature extraction from free-text product descriptions, as well as existing work on feature extraction from product specifications.

Feature Extraction from Product Descriptions: Several methods for extracting attribute-value pairs form product descriptions have been developed for the use case of product matching. The methods either use bag-of-words approaches to extract attribute-value pairs from the descriptions [3, 4, 12, 24, 25], a dictionary-based approach [6], or a combination of both [9, 18, 19].

In contrast, named entity recognition based feature extraction models are developed in [14, 17, 23]. All approaches use a similar models for feature extraction. In [14] propose an approach for annotating products descriptions based on a sequence BIO tagging model, following an NLP text chunking process. Specifically, the authors train a linear-chain conditional random field model on a manually annotated training dataset, to identify only eight general classes of terms. However, the approach is not able to extract explicit attribute-value pairs. Ristoski and Mika [23] improved upon this shortcoming employing a CRF model using a comprehensive set of discrete features that comes from the standard distribution of the Stanford NER[3] mode. Ortona et al. [17] propose a three fold approach that performs the following functions: validation of the offers values, blocking to reduce the number of compared offers, and scoring of the pairwise offers. For the validation, an annotator is used which performs NER extraction (places, locations, names, organizations), and ontology which contains some domain specific constrains. In the blocking step, all pairs of products that violate some of the ontology constrains are clustered in different clusters. In the third step, pairwise scores are calculated for the offers in each cluster.

Recently, several approaches employ word embeddings as additional knowledge for extracting features from product descriptions for the use cases of product matching [7, 26], product recommendation [5, 13, 28], and product classification [10]. However, the approaches can not bypass the problem of free-text product descriptions often covering only small number of features.

Feature extraction from Product Specifications: While there is a relatively large body of research for extracting product features from product descriptions, only a handful of works have studied the problem of feature extraction from semi-structured data within web pages such as HTML tables and HTML lists.

Etzioni et al. [2] relies on a approach from [8] to extract plane ticket prices from HTML tables within web pages. Specifically, the method involves automatically learning wrappers relaying on so called "landmarks" (i.e., groups of consecutive tokens) that enable a wrapper to locate the start and end of the item within the page. However, it is widely known that wrappers are inefficient way to solve the problem of automatic feature extraction from semi-structured data, since they have to be learned for every website.

The DEXTER system presented by Qui et al. [21] is the most prominent approach for extracting attribute-value pairs from HTML tables and HTML lists. The system is evaluated on the task of product data extraction. In order to identify HTML tables and lists within product web pages, DEXTER uses a binary classifier. To extract attribute-value pair, the system relies on straight forward heuristics. Namely, for extraction of specification tables the authors propose to extract the left column as the attribute name and everything else as a concatenated attribute value. For extraction of list specification the authors propose to split each list item by popular delimiters, and employ the above rule afterward.

An earlier work was presented by Wong et al. [27], where the authors build a graphical model, which employs latent Dirichlet process mixture model for extracting and aligning product attributes. To build the latent Dirichlet process mixture model, an unsupervised inference algorithm based on variational method is derived. This idea, was extended in [1], where the authors only focus on extracting "*positive*" features from specifications, by employing CRF model similar to [23]. The list of "*positive*" features is constructed by employing a sentiment analysis of product reviews found in the web page.

The authors of [16] perform product matching on a dataset of the Bing search engine. In their approach, the authors use historical knowledge to generate the attributes and to perform schema matching. In particular, they visit the merchant web page to compare the values of the products in the catalog with the values on the web page, converting the problem to a standard table schema matching problem. Next, similar to our approach the authors use instance-based schema matching to align the attributes' names in the catalog to the ones on the merchant web page.

4 METHODOLOGY

This section introduces our two step approach for extracting attribute-value pairs from technical specifications. Afterwards, we describe the matching method that we employ for aligning the product attribute schemata used by different vendors.

4.1 Product Feature Extraction

Automatically extracting specifications from HTML pages is not a trivial task. The technical specifications can be contained in different HTML structures, however they are primarily found in tables and lists [20, 21].

Similar to [21], we employ a two step approach: First, we detect specification tables and list in HTML pages: afterwards, we extract attribute-values pairs from the detected product specifications. We start by training a model for detection of specification tables and lists. Next, we apply that model on the web pages. Subsequently, we use a sample of the detected specifications to learn a model for column attribute and value detection. Finally, with the learned model we extract attribute value pairs from the specifications. The overall specification detection and attribute-value pair extraction process is shown in Figure 2.

[3] http://nlp.stanford.edu/software/CRF-NER.shtml

Specification Detection. As stated above product specifications are mostly found in tables and list. Considering that HTML tables and lists have a different base structure, we train separate classifiers for detecting specification tables and lists. The models are trained by learning a binary classifier that classifies tables/list into specification and non-specification.

The classifiers use the following features which have also been used previously by Qui et al. [21]: Average text length per row, number of rows, overall frequency of the word "specification", number of links and number of images, standard deviation of text size. In addition to these features, we introduce the following new features in order to improve the detection accuracy: Average DOM node depth of the items relative to the root, average number of columns, standard deviation of columns, maximum number of columns, number of non table/list tags, average ratio between numerical and alphabetical characters in a cell, and maximum number of rows. In order to illustrate the correlation of these features with the target attribute (specification vs. non-specification), Figure 3 shows summary statistics about the evaluation dataset that we use in Section 5. As it is evident from the Figure 3, there is a clear difference between specifications and non-specifications when considering these features. For instance, when considering "*standard deviation of columns*", specification tables hardly deviate from a given layout, while non-specification tables deviate form the layout much more. Another interesting feature is the "*number of non-table\list tags*" where non-specification structures contain much more tags like "*<p>, *" etc. than the specification ones. With that said, it is intuitive that the binary classifier could learn a better model if both feature sets are used.

Specification Extraction. The authors of [21] apply straight forward heuristics to extract attribute-value pairs from product specifications. Specifically, for tables they use a heuristic that, for each table row, extracts the first cell as attribute name and the remaining ones are extracted as concatenated values. Main limitation of this heuristics is that it can not handle tables that have more than two attribute name columns, like the one in Figure 1b. Namely, instead yielding a correct result like in Figure 4, this heuristic will treat values in the third column as attribute values, which is obviously incorrect . To solve this issue we train a model which can detect whether a given column is an attribute or a value column. Much like the detection model, we train a binary classifier with the following features: average text length per cell, average text length per column, number of non table tags per column, standard deviation of text size per column, ratio between alphabetical and numerical characters in a cell.

Taking into account that lists follow different structure (no columns), we convert list items to columns by separating the items by a delimiter and organise the separation result into columns. We consider the common delimiter characters: *":" and ";"*. We don't use delimiters like *","*, *"-"*, *"/" and "\"* since they might be a part of a product identification numbers like *MPN*. After the conversion process is done we are able to train the same model as described above. However, like in [21], this approach falls short when the lists do not contain any delimiter. Since the percentage of specification with out delimiters is less than 8%, we do not pursue a solution of this case.

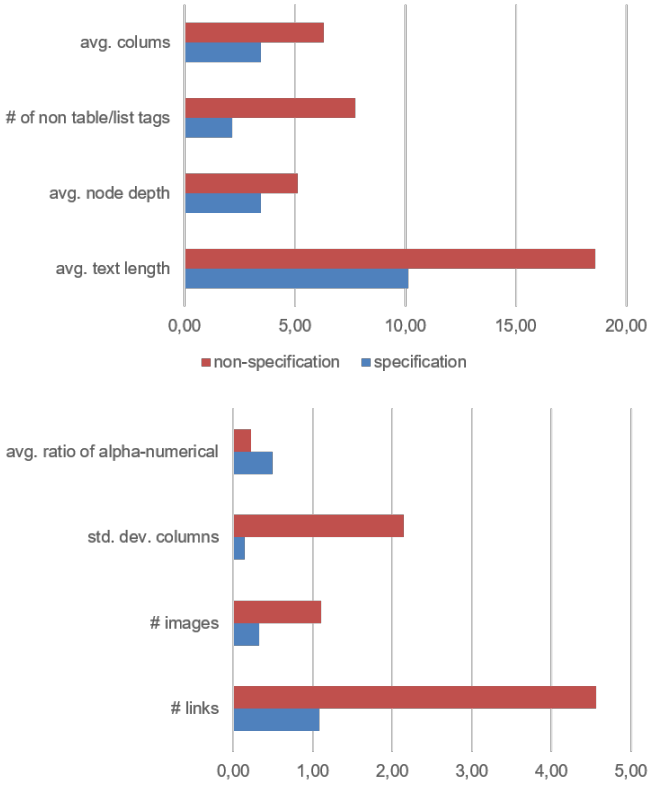

Figure 3: Specification vs. Non-specification table/list features

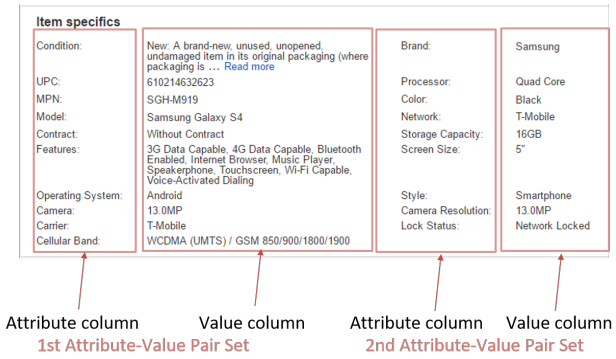

Figure 4: Example of extracted Product Specification Table

After the columns have been classified, we continue with pairing attribute with value columns, after which each attribute and value item, row wise, is considered a pair. The pairing of columns is done left to right, that is starting from the most left we pair the first attribute and value columns and we continue to the right. In the case of consecutive attribute or value columns, we concatenate them. Figure 4 shows an example of a table with tagged attribute and value columns, where the first and second column constitute

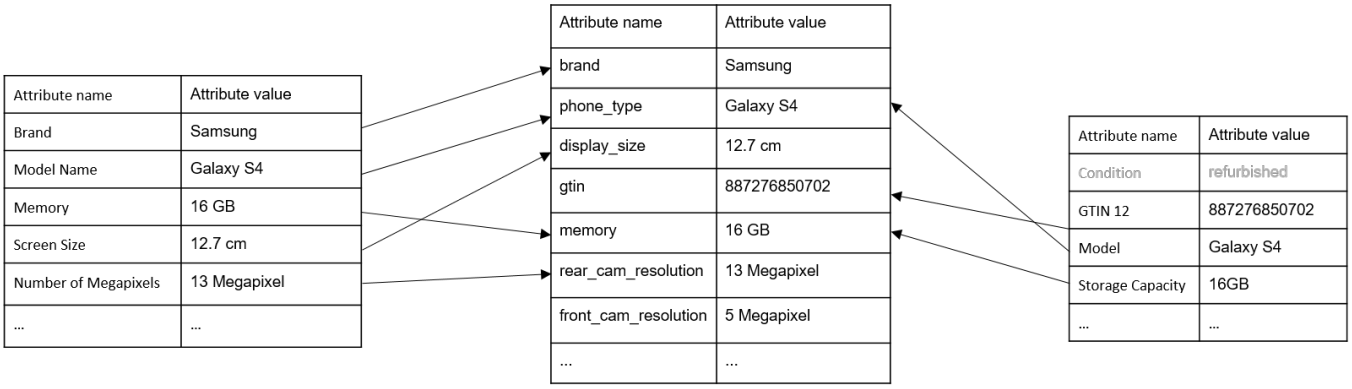

Figure 5: Examples of correspondences between product attributes: (left) Specification from walmart.com, (center) Centralised product Catalog and (right) Specification from ebay.com

the first attribute-value pairing, while the third and the fourth column constitute the second attribute-value pairing.

4.2 Product Schema Alignment

Given that vendors use different schemata to describe products, there is a need to align these schemata before tasks such as product matching, faceted product search, or content-based product recommendation can be executed.

Motivated by [16, 22], we employ a duplicate-based approach for matching the product attribute schema that are used by different e-shops. The prerequisite for being able to employ this approach is having access to a mid-sized amount of correspondences between offers for the same product by different e-shops. We use the correspondences between products to identify correspondences between the attributes that are used to describe the products. These product correspondences can be obtained using different methods, including exploiting shared identifiers such as GTIN, UPC, or EAN numbers or by relying on manual labeling. We use the Web Data Commons Gold Standard for Product Matching and Product Feature Extraction [20] (see Section 5.1), which contains *564* correspondences between product offers form three product categories (*headphones, phones and TVs*). The product offers originate from 32 web sites.

We start by assuming that same attributes have the same or very similar attribute values. This is why a core task of the schema matching method is to determine how similar two attribute values are.

We do not rely on exact equality of attribute values, since different specifications can have small variations for their attribute values. Moreover, attributes values do not have explicit data types, making an effective determination of similarity between these values considerably more difficult. After careful analysis of the the extracted attributes, we end up with three data types: categorical (strings), numerical and units of measurements. Since attribute values often contain only one data type, prediction of that type is a straightforward task. For the data type detection we use the heuristic introduced in [11]. The data type of each attribute is identified based on its values. First, the data type of each value of the attribute is detected by using about 100 manually defined regular expressions which are able to detect the data types number (with or without unit of measurement). Additionally, there are around 200 manually generated rules for converting units of measurements to the corresponding base unit.

In the following we explain the similarity measures for each data type:

Categorical (strings): We employ Soft-Jaccard similarity on word-level n-grams (n=(2,3)), where we consider values as similar if the threshold $\theta > 0.822$ (the threshold is set by running exhaustive search on the possible pairs).

Numbers: A simple and in our experiments effective approach is to calculate the ratio between the absolute values of the numbers. As with strings, numbers can have small variations because of typos or rounding (e.g. 5 vs. 4.7). Therefore, we consider to number to be similar if the threshold $\theta > 0.901$ (the threshold is set by running exhaustive search on the possible pairs).

Unit of measurements: Product specifications often use different units of measurements (metric vs imperial). This can make simple number comparison ineffective. For that purpose, we first convert every unit of measurement to the metric system and then employ the numeric comparison.

Having defined the similarity measures, we continue with applying them to each attribute pair with compatible data types. Similar to [22], we apply a straightforward approach to match the two schemata. For each attribute in both product specifications, the match with the best score is determined and the final mapping is adopted only if an attribute a from the first product specification has an attribute b as their best match and vice versa, i.e we only consider a match if the best scores are bi-directional. Finally, the approach yields a set of bi-directional schema correspondences sorted per attributes in C. Figure 5 shows examples of extracted specifications together with discovered correspondences between product attributes.

5 EVALUATION

In this section we provide the evaluation for both the feature extraction from technical specification and schema alignments methods. We first detail the datasets used, highlighting some of the most

Table 1: Number of product offers per category in the WDC Gold Standard

Category	# of Offers
Headphones	156
Phones	254
TVs	154

interesting properties used within the dataset. Next, we describe the experiment setup. Finally we discuss the results.

5.1 Datasets

We use the Web Data Commons (WDC) Gold Standard for Product Matching and Product Feature Extraction [20][4]. The gold standard consists of product offers from 32 different websites. For each offer, all product features have been manually annotated which appear in: (i) the name of the product marked up using the Microdata syntax, (ii) description of the product marked up with Microdata, (iii) specification tables, and (iv) specification lists.

Product Feature Extraction Dataset. The gold standard for product feature extraction contains a set 564 product offers found on the web, with annotated attributes found in specification tables and/or list. The offers were chosen from three categories: Headphones, Phones and TVs. Table 1 shows the number of offers by category. From this set, for the purposes of the specification detection evaluation, we used 334 specification tables and 169 specification lists. Additionally we annotated 304 non-specification tables (layout, listing etc.) and 150 non-specification lists as negative examples.

Table 2 shows the product attribute densities for both the *Attribute Density in Descriptions* (ADD) (Product Title and Text) and *Attribute Density in Specifications* (ADS). Since there are more than 30 attributes per category only a sample of ten attributes per category is shown. Generally, attribute density for ADD is lower than for ADS, i.e attributes can be found more often in product specifications than in product descriptions. Moreover, the table shows that some attributes could not be found in the product descriptions at all, whereas they can be found in product specification.

Product Schema Alignment Dataset. We use the same dataset used for the product feature extraction, with the addition of correspondences to a unified product catalog containing 150 products from the following categories: headphones (50), phones (50), and TVs (50). The attributes in the catalog were scraped from leading shopping services, like Google Shopping, or directly from the vendor's website. In total, the data set contains 564 correspondences between product offers from the 32 e-shops and the central catalog.

5.2 Experiment Setup

The models for table and list detection as well as the column detection were trained and evaluated with 5 cross-fold validation. We used a SVM classifier implementation provided within the libSVM[5]. The SVM model was built with a linear kernel to train both the detection and extraction models.

[4]http://webdatacommons.org/productcorpus/
[5]https://www.csie.ntu.edu.tw/ cjlin/libsvm/

Table 2: Densities of the attributes within the Product Feature Extraction Gold Standard. *Attribute Density in Descriptions* (in the following *ADD*) represents the attribute density of attributes found in the product description, while *Attribute Density in Specifications* (in the following *ADS*) represents the attribute density of attributes found in the product specifications.

Attribute	ADD %	ADS %
Headphones		
Brand	84	97
Product Name	81	87
MPN	N/A	81
Color	39	56
Sensitivity	N/A	53
Impedance	10	53
Cup Type	N/A	47
Form Factor	24	43
Magnet Mat.	2	27
Diaphragm	N/A	25
Phones		
Product Name	81	91
Memory	73	87
Brand	87	86
Color	71	79
Display Size	18	71
Rear Cam. Res.	10	70
OS	11	64
Display Res.	6	48
Processor	11	28
Front Cam. Res.	1	20
TVs		
Brand	79	100
Product Name	67	91
Display Type	70	81
Display Size	50	65
Display Res	20	55
Tot. Size	45	51
Ref. Rate	20	50
Img. Asp. Rat.	3	38
Connectivity	2	35
Resp. Time	N/A	10

The schema correspondences for the schema alignment task were inferred from 70% of the correspondences and tested on the rest of the data[6].

5.3 Results

Product Feature Extraction. Table 3 gives results on table detection. As a baseline for the experiments we use DEXTER introduced in [21]. Our approach outperforms DEXTER in F-score by 5%, as a result of adding more subtle features to our classifier. On the other

[6]The models, data and code for both feature extraction and schema alignment can be found at http://webdatacommons.org/productcorpus/

Table 3: Detection Results for HTML Tables and Lists

Tables			
	Precision	Recall	F-score
DEXTER[21]	0.820	0.887	0.842
Our approach	0.904	0.883	**0.892**
Lists			
	Precision	Recall	F-score
DEXTER[21]	0.601	0.604	0.602
Our approach	0.623	0.620	0.621

Table 4: Extraction Results for HTML Tables and Lists

Tables			
	Precision	Recall	F-score
Dictionary[20]	0.560	0.609	0.583
DEXTER[21]	0.691	0.778	0.724
Our approach	0.800	0.826	**0.817**
Lists			
	Precision	Recall	F-score
Dictionary[20]	0.414	0.527	0.463
DEXTER[21]	0.548	0.623	0.583
Our approach	0.700	0.556	**0.619**

Table 5: Product Schema Alignment results per Category

	Precision	Recall	F-score
Headphones schema	0.946	0.967	0.956
Phones schema	0.885	0.958	0.920
TVs schema	0.862	0.953	0.905

hand, as a result of not using additional features for the list classifier, our approach for list detection does not show any significant improvement over the baseline.

Tables 4 shows the comparative results on the attribute-value pair extraction for tables and list against the baselines. In particular, we compare to: (1) the dictionary approach presented in [20] and (2) the approach introduced in [21] (DEXTER). Comparably, list specifications proved to be more difficult to extract than tables. However, in both cases our approach outperforms the DEXTER and the dictionary approach baselines. In the case for table specification extraction by almost 10% and in the case of lists by 3%. The significant increase in F-score for table specification can be explained by the fact that our model can extract attribute-value pairs from tables which have more than a single attribute column. For instance, when extracting attribute-value pairs from specification like in Figure 4 with the DEXTER approach the first column would be extracted as attributes and all other would be concatenated values introducing *false positives* and moreover not being able to extract the third and fourth column as separate attribute-value pairs introduces *false negatives*. On the other hand, list attribute-value pair extraction still remains difficult task, since: (i) the detection of product specification within lists is a more difficult task, and (ii) a small number of the lists in the used dataset do not contain any kind of delimiters.

Product Schema Alignment. The result of our schema alignment experiment is shown in Table 5. Our duplicate-based matching methods works quite well for all three product categories (all three F-scores are above 90%). Noteworthy, is that for the headphones category we get significantly higher F-score. This is a result of the consistency of the attributes with which headphones are described by the e-shops, as well as the lower number of distinct attributes in general. Contrary to the the headphones category, the phones and TVs categories both have larger number of distinct attributes and therefore attributes in these categories are not consistently used to describe them.

6 CONCLUSION

Attribute-value pair extraction is a crucial prerequisite for many e-commerce tasks including product matching, recommendation and categorisation. However, product names and descriptions often only cover a relatively small number of product attributes and it is thus beneficial to extract product features from product specifications which often cover a much larger number of features. In this paper, we have proposed an approach for extracting attribute-value pairs from product specifications. To perform the detection, and handle the high diversity of specifications in terms of content, size and format, our approach uses supervised learning to classify HTML tables and lists present in web pages as specifications or not. To perform extraction of the attribute-value pairs from the HTML fragments identified by the specification detector, we again use supervised learning to classify columns as an attribute or value column. We show improvements in F-score over the state-of-the-art [21] of up to 10% in extracting attribute-value pairs from tables and up to 3% for lists.

There are a number of potential future research directions. One could certainly try to improve list specification detection and extraction by introducing better features for the detection and extraction models. Additionally, with the of rise HTML5 more and more specifications are embedded into other layout friendly elements, thus increasing the potential for finding features in those elements.

REFERENCES

[1] Lidong Bing, Tak-Lam Wong, and Wai Lam. 2016. Unsupervised Extraction of Popular Product Attributes from E-Commerce Web Sites by Considering Customer Reviews. *ACM Trans. Internet Technol.* 16, 2, Article 12 (April 2016), 17 pages. DOI: http://dx.doi.org/10.1145/2857054

[2] Oren Etzioni, Rattapoom Tuchinda, Craig A. Knoblock, and Alexander Yates. 2003. To Buy or Not to Buy: Mining Airfare Data to Minimize Ticket Purchase Price. In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '03).* ACM, New York, NY, USA, 119–128. DOI: http://dx.doi.org/10.1145/956750.956767

[3] Rayid Ghani, Katharina Probst, Yan Liu, Marko Krema, and Andrew Fano. 2006. Text mining for product attribute extraction. *ACM SIGKDD Explorations Newsletter* 8, 1 (2006), 41–48.

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

[5] Mihajlo Grbovic, Vladan Radosavljevic, Nemanja Djuric, Narayan Bhamidipati, Jaikit Savla, Varun Bhagwan, and Doug Sharp. 2015. E-commerce in Your Inbox: Product Recommendations at Scale. In *Proceedings of the 21th ACM SIGKDD.* ACM, 1809–1818. DOI: http://dx.doi.org/10.1145/2783258.2788627

[6] Anitha Kannan, Inmar E Givoni, Rakesh Agrawal, and Ariel Fuxman. 2011. Matching unstructured product offers to structured product specifications. In *17th ACM SIGKDD.*

[7] M. H. Kiapour, X. Han, S. Lazebnik, A. C. Berg, and T. L. Berg. 2015. Where to Buy It: Matching Street Clothing Photos in Online Shops. In *2015 IEEE International*

Conference on Computer Vision (ICCV). 3343–3351. DOI: http://dx.doi.org/10.1109/ICCV.2015.382

[8] Craig A. Knoblock, Kristina Lerman, Steven Minton, and Ion Muslea. 2003. *Accurately and Reliably Extracting Data from the Web: A Machine Learning Approach*. Physica-Verlag HD, Heidelberg, 275–287. DOI: http://dx.doi.org/10.1007/978-3-7908-1772-0_17

[9] Hanna Köpcke, Andreas Thor, Stefan Thomas, and Erhard Rahm. 2012. Tailoring entity resolution for matching product offers. In *Proceedings of the 15th International Conference on Extending Database Technology*. ACM, 545–550.

[10] Zornitsa Kozareva. 2015. Everyone Likes Shopping! Multi-class Product Categorization for e-Commerce. In *The 2015 Annual Conference of the North Americal Chapter for the ACL*. 1329–1333.

[11] Oliver Lehmberg, Dominique Ritze, Petar Ristoski, Robert Meusel, Heiko Paulheim, and Christian Bizer. 2015. The Mannheim Search Join Engine. *Web Semantics: Science, Services and Agents on the World Wide Web* 35 (2015), 159 – 166. DOI: http://dx.doi.org/10.1016/j.websem.2015.05.001 Semantic Web Challenge 2014.

[12] Nikhil Londhe, Vishrawas Gopalakrishnan, Aidong Zhang, Hung Q Ngo, and Rohini Srihari. 2014. Matching titles with cross title web-search enrichment and community detection. *Proceedings of the VLDB Endowment* 7, 12 (2014), 1167–1178.

[13] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference*. ACM, 43–52.

[14] Gabor Melli. 2014. Shallow Semantic Parsing of Product Offering Titles (for better automatic hyperlink insertion). In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 1670–1678.

[15] Robert Meusel, Petar Petrovski, and Christian Bizer. 2014. The webdatacommons microdata, RDFa and microformat dataset series. In *The Semantic Web–ISWC*. 277–292.

[16] Hoa Nguyen, Ariel Fuxman, Stelios Paparizos, Juliana Freire, and Rakesh Agrawal. 2011. Synthesizing products for online catalogs. *Proceedings of the VLDB Endowment* 4, 7 (2011), 409–418.

[17] Stefano Ortona. 2014. An analysis of duplicate on web extracted objects. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*. 1279–1284.

[18] Petar Petrovski, Volha Bryl, and Christian Bizer. 2014. Integrating product data from websites offering microdata markup. In *Proceedings of the companion publication of the 23rd international conference on World wide web companion*. 1299–1304.

[19] Petar Petrovski, Volha Bryl, and Christian Bizer. 2014. Learning Regular Expressions for the Extraction of Product Attributes from E-commerce Microdata. (2014).

[20] Petar Petrovski, Anna Primpeli, Robert Meusel, and Christian Bizer. 2017. *The WDC Gold Standards for Product Feature Extraction and Product Matching*. Springer International Publishing, Cham, 73–86. DOI: http://dx.doi.org/10.1007/978-3-319-53676-7_6

[21] Disheng Qiu, Luciano Barbosa, Xin Luna Dong, Yanyan Shen, and Divesh Srivastava. 2015. Dexter: large-scale discovery and extraction of product specifications on the web. *Proceedings of the VLDB Endowment* 8, 13 (2015), 2194–2205.

[22] Daniel Rinser, Dustin Lange, and Felix Naumann. 2013. Cross-lingual Entity Matching and Infobox Alignment in Wikipedia. *Inf. Syst.* 38, 6 (Sept. 2013), 887–907. DOI: http://dx.doi.org/10.1016/j.is.2012.10.003

[23] Petar Ristoski and Peter Mika. 2016. Enriching Product Ads with Metadata from HTML Annotations. In *Proceedings of the 13th Extended Semantic Web Conference. (To Appear)*.

[24] Ronald van Bezu, Sjoerd Borst, Rick Rijkse, Jim Verhagen, Damir Vandic, and Flavius Frasincar. 2015. Multi-component Similarity Method for Web Product Duplicate Detection. (2015).

[25] Damir Vandic, Jan-Willem Van Dam, and Flavius Frasincar. 2012. Faceted product search powered by the Semantic Web. *Decision Support Systems* 53, 3 (2012), 425–437.

[26] Xi Wang, Zhenfeng Sun, Wenqiang Zhang, Yu Zhou, and Yu-Gang Jiang. 2016. Matching User Photos to Online Products with Robust Deep Features. In *Proceedings of the 2016 ACM on International Conference on Multimedia Retrieval (ICMR '16)*. ACM, New York, NY, USA, 7–14. DOI: http://dx.doi.org/10.1145/2911996.2912002

[27] Tak-Lam Wong, Wai Lam, and Tik-Shun Wong. 2008. An Unsupervised Framework for Extracting and Normalizing Product Attributes from Multiple Web Sites. In *Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '08)*. ACM, New York, NY, USA, 35–42. DOI: http://dx.doi.org/10.1145/1390334.1390343

[28] W. X. Zhao, S. Li, Y. He, E. Chang, J. R. Wen, and X. Li. 2015. Connecting Social Media to E-Commerce: Cold-Start Product Recommendation On Microblogs. *IEEE Transactions on Knowledge and Data Engineering* PP, 99 (2015), 1–1. DOI: http://dx.doi.org/10.1109/TKDE.2015.2508816