

Automatic Extraction of Attributes and Entities for Product Differentiation

Abhijeet Ramesh Thakare *, Parag S. Deshpande

*Department of Computer Science and Engineering, Visvesvaraya National Institute of Technology
South Ambazari Road, Nagpur, Maharashtra 440010, India*

E-mail: abhijeet.thakare@gmail.com, psdeshpande@cse.vnit.ac.in

Received 30 April 2017

Accepted 28 October 2017

Abstract

This paper introduces a novel concept of web-spreadsheet, which extracts product information by crawling through related web pages and generates information like a spreadsheet where each row represents product information and each column represents product attributes. Using Decision Tree based classifier, the accuracy achieved for extracting entities and attribute-value pairs is 98.71% and 99.99% respectively. This strategy will benefit the product companies for performing successful product differentiation and save a lot of time of the customers for comparing different products of same product class.

Keywords: Attribute value extraction, Entity extraction, Product Differentiation, Web Spreadsheet, Aggregated Search

1. Introduction

Rapid development of e-commerce has brought vast opportunities to the customers as well as product manufacturers or retailers. The main objective of each product manufacturer is to highlight the special features of the product. Product differentiation is a marketing technique which depicts the differences between the products. Differentiation makes the product more interesting by comparing its unique qualities with other competing products in the market. Successful product differentiation helps product manufacturers to gain competitive advantage in the market. It also helps the customers to choose the right product of their choice. The major factors on which product differentiation can be performed are price, features, quality and availability (time and location) of the product. In this paper, we focus on extracting automatically entities, price and features of the product for the specified product class.

Currently, customers visit different manufacturer or retailers site for purchasing the product. For a particular class of a product, only a limited amount of information is available on a single website regarding the product features. Customers will also find different values of the features as well as prices for the product on various websites. Therefore, the customer has to visit numerous websites to choose the right product. If any retailer or manufacturer has introduced new product feature, the feature information is manually updated on his website. Some websites display a comparison of similar class products. But, this comparison process lacks new feature introduced by another manufacturer.

For carrying out fruitful product differentiation, a web-spreadsheet view is a solution which will handle all above discussed scenarios. Hence, in the proposed research, we focus on novel search problem: Automatic Extraction of Attributes, their corre-

* Corresponding author.

sponding values and Entity Information (AEAVEI). The main objective of AEAVEI is to extract relevant attribute-value information related to entities along with the entity (product) name of the specified product class in the form of a spreadsheet. Web pages are crawled for extraction of attributes and their respective value-measure pairs. Extracted attributes will form the columns of the spreadsheet and values (value-measure pair) along with entity name will form rows of a spreadsheet. This approach provides all product information for particular product attributes under one roof.

AEAVEI is inspired by Relational aggregated Search (RAS). On the web, large amount of relations exists between the information. The goal of the RAS is to find out these relations between the information, extract and aggregate this information in the form of small nuggets. RAS consists of three main parts, namely Query Dispatching (QD), Nuggets Retrieval (NR) and Result Aggregation (RA) [1]. In the proposed work, we mainly focus on relation retrieval between information placed on the web and aggregating the information in spreadsheet format. In relation retrieval, we focused on instance-attribute relation to automatically extract attributes and their respective values and instance-class relation to automatically extract entity (product) name of a particular class. For example, if a user submits a query for the product class Mobile Phone, then the entities belonging to the class Mobile-Phone and attributes and their values of those entities are extracted.

By applying this approach, the effort of user by crawling through each hyperlink of product pages for finding relevant information is reduced. This approach also aggregates updated product information (attribute and value-measure pairs) dynamically in a web-spreadsheet.

Aggregating information in a consolidated spreadsheet is having following challenges:

- On the web page, attributes and their values are not directly mentioned, so correct pairs of attribute and their values has to be extracted.
- On the web page, descriptions of entities are generally stored in an unstructured manner. The product (entity) name may appear on a web page in an ambiguous manner. For example, when user in-

tention is electronic tablet, it may search webpage of medicinal tablet.

1.1. Contributions

Research work is carried out by considering the above challenges and the major contributions are summarized as follows:

- A novel attribute-value extraction method is designed to extract attributes and their respective values from unstructured text which works independently of the arrangement of the attribute and its value.
- A novel complete entity name extraction process is proposed to extract complete entity name from unstructured text.
- Generating web-spreadsheet dynamically without referring any additional information, and irrespective of domain without targeting some specific set of retailer websites.
- As per our knowledge, no other system is designed which automatically extracts entities, and their respective attribute-value pairs for specific product class as we have conveyed in this paper. We have crawled almost all related web pages to efficiently extract entities and attribute-value pairs.
- The proposed method for entity and attribute-value extraction achieves very good accuracy which is measured in terms of precision, recall and F1-measure that is superior to eminent entity and attribute value extraction methods discussed in the literature.

The rest of the research work is organized as follows. In Section 2, related studies for extracting named entity and Attribute-Value are discussed. Section 3 describes proposed work which discusses definitions and terms and proposed algorithm in detail. Section 4 discusses softwares and hardware used for carrying out experiments. In Section 5, experimental results of the proposed system developed are shown. The conclusions and future work of the paper are given in Section 6.

2. Related work

Extracting information from unstructured text that is natural language documents have been proposed [2, 3]. A template independent wrapper has been developed which exercises labeled news pages from a single site [4]. This wrapper has been used to extract plain texts of news embedded in news article body. The relation between news article and news body is also explored by developing novel features for them. Various text summarization techniques has been discussed to summarize unstructured documents like news articles, nursing narratives and country economic reports [5]. There are some approaches that deal with semistructured documents [6]. In these documents, information is stored in different HTML structures like tables, list and forms. Using ORASS (object-relationship-attribute model), semantics has been efficiently extracted from semistructured data of web documents [7]. The task of designing semistructured databases will become easy using ORASS model.

Extracting entities from web pages is a challenging problem, because entities are dispersed over web pages in an ambiguous and unstructured manner. Nowadays, by identifying instance-class relation to extract entity (instance) from web pages is also demanding. Using predicate and patterns, named entities are extracted from web corpus [8]. In this work, using Bootstrapping a set of rules are created which utilizes domain independent extraction patterns. Accuracy is calculated using mutual information between extracted instances of a class and set of discriminator associated with that class. Various techniques based on ontologies have been applied for enhancing the performance of information management systems. Ontology is one of the best techniques to represent background knowledge in information management systems. Background knowledge or domain knowledge such as likings of the user is very necessary for understanding situations and problems in Information management systems. To understand mutual monitoring for enhancing recommendation quality in autonomous multi-agent systems, [9] focused on ontologies and ANNs to serve as behavior and interests of users.

In [10], the framework named as Artequakt has

been developed. Artequakt utilizes ontology relation declaration and lexical data to identify a relation between entities. These entities are stored in unstructured web documents. In this work initially, an ontology for the artists and paintings has been created. Using various information extraction (IE) tools and techniques, ontology is automatically populated with the information extracted from web documents by utilizing ontology representation and WordNet lexicons. The information is then stored in the knowledge base (KB). Afterwards, a query is fired to KB to search and extract the relevant text or fact. This text or fact is used to generate biography dynamically. A technique for finding semantic associations between web entities by considering the complex hierarchical structure of web entities has been proposed in [11]. In this work, additional information such as a logical collection of web entities into groups has been considered for extracting semantic association between web entities. Experimentation has been performed by considering MUADDIB [12, 13] recommender systems.

Work described in [14] addresses the technique of extracting and grouping the words or entities from the text into pre-defined domain-specific concepts. This technique utilizes parts of speech (POS) tagging. For the extraction purpose, Engineering Ontology from the mechanical domain has been taken into consideration. For automatic extraction of entities, data from various online sources which depict mechanical components have been taken into account. In this work, four domain, i.e. concepts, device, material, and a process has been automatically extracted.

Automated feature induction method has been proposed [15]. In this method, Named entity recognition is done using undirected graphical models called as Conditional Random Fields (CRFs). Initially, the user creates atomic features. In this work, as compared to traditional approaches accuracy is improved. This method also reduces feature count. CONLL-2003 named entity shared task dataset has been used for experimentation.

Focusing on context patterns for extracting named entity from unlabeled data has been suggested [16]. This work utilizes language indepen-

dent technique. The initial seed list is created and it is gradually increased by searching more instances for matching context patterns from unlabeled data.

Work addressed in [17], Support Vector Machine (SVM) based parser has been presented. N-best words with the features based on their character position and type has been calculated using statistical techniques for an input sequence. Finally, the characters are aggregated into words using SVM and named entity is recognized. In this work, CRL NE data have been used for evaluation. A System developed for extracting named entities from unstructured biomedical documents using integrated SVM and Decision Tree classifier has been proposed in [18].

In the past few years, extracting attributes and their values from web pages had gained the attention to researchers. Extracting implicit and explicit attributes from the textual product description has been proposed [19]. Implicit attributes are not directly mentioned in the product description. They are semantic attributes. While, explicit attributes are physical attributes. Using Naïve Bayes as a supervised learning and Co-training Expectation Maximization (Co-EM) as a Multiview semi-supervised learning algorithm, explicit attributes are extracted. In this work, sporting goods category website has been used for extracting attribute values.

Work described in [20], extracted textual attribute value pairs under three diversities such as Tweets, Web-Links and Tweets+ Web-Links for 15 events. By using set of delimiters, sentences are split and attribute-value pairs are extracted. Some special cases are handled using common noun or proper noun. Remaining sentences which doesn't fall under special cases are handled using dependencies (subject and object concept).

Work on extracting attribute-value pairs using Knowledge Base (KB) for a particular category have been demonstrated [21]. In this work, unstructured web pages are used for creation of KB. Table and other structures available in unstructured pages are used for automatic creation of KB. Using this KB, web pages are annotated automatically. These annotated pages are utilized for building training model for extraction. Some attribute value pairs are miss-

ing in this model because, annotated corpus is not created manually. Therefore, from unstructured test data, the model fails to extract some attribute-value pairs.

Recently lot of work has been done for extracting attributes and their values from online reviews of the product. In the work done by [22], features are extracted from the pros and cons sentences of the customer views of the product. For extracting product features and opinions from reviews, [23] calculate point of mutual information between the noun-phrases and related context patterns. A Technique based on association rule mining and natural language processing for mining frequent features has been proposed [24].

3. Proposed work

Following subsections describe proposed work. These subsections present the meaning of definitions and terms used in the Algorithm 1, important steps of the algorithm and explanation of each step of the algorithm. The attribute-value extraction which is given in Algorithm 1 is implemented as a two step process. In the first step entity name is extracted [25] and after entity name extraction, attribute-value pairs are extracted in a second step.

3.1. Definitions and terms

Class Name (CLSNAME): CLSNAME is a product class name which is provided by the user. For example, value of CLSNAME can be Washing Machine, Tablet or Tennis Racket etc. Our final aim is to generate a spreadsheet for class CLSNAME.

Web Spreadsheet (WSP): WSP is a spreadsheet which consist of $\{R_1, R_2, R_3, \dots, R_n\}$ rows. Consider, W be a set of web pages related to product class, where, $W = \{W_0, W_1, W_2, \dots, W_n\}$ and $n \geq 0$. Algorithm 1, crawls over each W_i one by one and generates one row T_i for spreadsheet WSP. We define function $f: W \rightarrow T$ where, f is injective and onto.

Web Spreadsheet Columns (WCL): WCL consists of $\{A_1, A_2, \dots, A_n\}$. attributes. These attributes are automatically extracted by Algorithm 1.

Web Spreadsheet Row (R_i): It is one of the rows of a spreadsheet. R_i consists of $(E_i, V_1, V_2, \dots, V_n)$

Algorithm 1 Entity and Attribute-Value Extraction

- 1: **Input:** Search Query specifying class name CLSNAME, Price as a single attribute and maximum number of URLs crawled (n) to limit the search space.
 - 2: **Output:** Spreadsheet (WSP) indicating product names of the class and attribute values represented as rows of a spreadsheet.
 - 3: Prepare Synonym list (SYL) for product class name CLSNAME.
 - 4: Extract URLs from the search engine by specifying classname CLSNAME and Price as an attribute and store them in URLSTORE.
 - 5: Rank the documents pointed by URL based on presence of classname and some of the attribute names belonging to class CLSNAME. Sort URLSTORE based on rank.
 - 6: **for** every document W_i present in URLSTORE **do**
 - 7: Remove Stop words, unwanted symbols and perform stemming using Word Net Stemmer.
 - 8: For each word in a sentence construct Feature Vector specified in sections "Extraction of entity name" and "Attribute-Value extraction"
 - 9: Classify word using Decision Tree Classifier for determining entity name (E_i)
 - 10: Classify word using Decision tree Classifier to determine Attribute-Value pair ($A_1 - V_2, A_2 - V_2, \dots, A_n - V_n$).
 - 11: Prepare Columns of spreadsheet using (A_1, A_2, \dots, A_n) attributes which is obtained in step 10.
 - 12: Prepare row of spreadsheet using results from step 9 and values (V_1, V_2, \dots, V_n) obtained in step 10.
 - 13: **end for**
 - 14: Output: spreadsheet(WSP)
-

where, E_i is a complete entity name and V_1, V_2, \dots, V_n are value-measure information for attributes A_1, A_2, \dots, A_n respectively corresponding to document W_i where, $n, i \geq 1$

POS tag of a word WD_i (POS_Tag): POS tag of word WD_i is defined as POS_Tag. POS tag of word is used to categorize the word as nouns, adjectives, verbs, adverbs etc. Following POS tags are used in the proposed work:

CD- Cardinal Number, NN -Noun, singular or mass, NNP - Proper noun, singular.

Synonym List (SYL): Synonym list for class, CLSNAME consists of set synonym words of CLSNAME. SYL is represented as $SYW_1, SYW_2, \dots, SYW_n$, where $n \geq 1$.

URLSTORE: List to store all related URL, returned by a search engine.

3.2. Input

Algorithm 1 takes input as search query containing product class name (CLSNAME), one of the attribute as price and maximum number of URLs (n) which can be crawled to limit the search space. For example, to prepare a spreadsheet for Mobile Phone user can specify product class name as "Mobile Phone", attribute as price and maximum number of URLs crawled as n .

In the proposed work, we define following two data types. These data types are used to mark "value-measure pair word" (VM). These data types are as follows:

Alphanumeric (AN): If the VM word consists of at least one Alphabetic letter (A-Z or a-z) and at least one digit (0-9) or combination of digit (0-9) and alphabetic letters then the word falls under alphanumeric category. For e.g. if we take camera as an

attribute and its value as 13MP then type of value of the camera is Alphanumeric.

Numeric-Alpha (NA): Numeric-Alpha category is a subset of Alphanumeric category. The only difference is value and measure is separated by one or more spaces. i.e. Number/s-Space-Alphabet/s (String). For e.g. 1 MP.

3.3. Output

Algorithm 1 generates spreadsheet for Product class name (CLSNAME) which consists of, columns as attribute names and rows an entity (product) name along with values for the corresponding attribute names. Attributes and their respective values as well as entity names are automatically extracted by the Algorithm 1.

3.4. Preparation of synonym list

We also maintain Synonym List (*SYL*) for product class name for which the spreadsheet has to be generated. All the synonym words for product class name are preserved in this list. Synonym list for class CLSNAME is created using Word Net. Synonym list plays a crucial role in extracting instance-class relation. The main aim of creating a synonym list is to target only those sentences of a web page, which contains at least one of the synonym words from list *SYL*.

Every Retailer describes their products in a different fashion. They also use different words corresponding to class name CLSNAME to describe the complete name of the product. For e.g., Figure 1 shows the initial part of product description page for Lenovo Tablet. In this, complete name (Lenovo Tab 2 A8) in the heading part comprises word "Tab" while in Figure 2, the complete name (UBI Slate 7CZ Tablet) consists of word "Tablet".

While describing the product, retailers use different substring words. These substring words which are basically a part of a complete entity name. For e.g., Figure 3 shows the part of product description page for the Samsung Galaxy Tablet. The Complete Name of the product is "Samsung Galaxy Tab 4". The product is described using n number of sentences. Consider the two sample sentences de-

scribed as follows:

Sentence 1: "Packed with a host of features, Samsung Galaxy Tab 4 T231 Tablet is a user friendly budget device."

Sentence 2: "Android operating system and 1.5 GB RAM boosts the performance of this Galaxy Tablet." In sentence 1, complete entity name, i.e. "Samsung Galaxy Tab 4 T231" is used while in sentence 2, substring word "Galaxy" is used. In sentence 1, "Tablet" and "Tab" are used as a synonym words while in sentence 2 "Tablet" is used as a synonym word.

3.5. Module for extraction of URL

Using jsoup 1.8.2 Java library, each URL in the search result which is returned by search query is accessed, parsed and decoded. All child URLs from each search result URL are also extracted, parsed and decoded. Iteration through every URL page and their child URL pages are done and the complete decoded URLs for these pages are stored in a database table. To ensure that, the URL is not visited twice, the presence of a URL in the database table is checked before storing it. If it is not present in the database table, then, the URL is stored into the table as well as in list URLSTORE. Now, the database table as well as list URLSTORE will consist of unique URLs. These URLs will be given as input to the module Data Preprocessing.


3.6. Documents ranking

The extracted documents are ranked based on presence of product class name and some attribute names of that particular class CLSNAME. The rank of the document is decided by a number of attributes present in the documents. Higher rank indicates more attributes are present in the document. If the same number of attributes are present in the document, then document which is having more priority attributes are ranked as higher.

3.7. Preprocessing of data

In a preprocessing of data, all stop words and special symbols are removed.

Lenova Tab 2 A8 (4G + Wifi, Calling, Midnight Blue)


84 Ratings, 8 Reviews, Q&A

- 1 Year Manufacturer Warranty
- Storage: 16 GB
- Screen Size(in cm): 20.32 cm
- Voice Call: Yes
- RAM: 1 GB


[View all item details](#)

Rs. 15,000 **Rs. 11,276** (25% off)

EMI starts at Rs.546

Fig. 1. Initial Part of Product Description Page for Lenovo Tablet.

UBI Slate 7CZ Tablet (7 inch, 8GB, Wi-Fi+3G+Voice Calling), Black


168 customer reviews, 43 answered questions

~~Rs. 5,999~~ **Rs. 3,216**

EMI starts at Rs.287.24 per month.

In Stock

Fig. 2. Initial Part of Product Description Page for UBI Slate Tablet.

Packed with host of features, **Samsung Galaxy Tab 4 T231** tablet is user friendly budget device. It comes with a large 17.78 cms (7) WXGA multi-touch display on which you can view images and videos with a resolution of 1280 * 800 pixels. With 3 megapixel rear camera of this tablet, you can capture stunning shots. It comes with zoom features that less you bring your subject closer to the camera lens. You can take amazing selfies with its 1.3 megapixels front camera. Equipped with 1.2 GHz quad-core processor, it offers responsive navigation across different applications. Android operating system and 1.5 GB RAM boosts the performance of this **Galaxy tablet**. On buying this **Samsung Galaxy tablet**, you not only get lowest price guarantee but also 1 Year Samsung India warranty on the product.

Fig. 3. Product Description Page for Samsung Galaxy Tablet.

3.8. Extraction of entity (product) name

3.8.1. Classifier construction for entity name using J48 decision tree [Weka]

Feature vector for each word in a sentence is constructed which is a major step in designing a classifier. Data is prepared for training and using decision tree, word is classified.

3.8.2. Building a features vector for each word

Each document W_i is crawled one by one. The URL of each W_i is already stored in the list URLSTORE as mentioned in the section 3.5 . Feature vector (Training data) is constructed using Tablet, Washing Machine and Tennis Racket classes by crawling the related website URL of different manufacturers or retailers. Every word in document W_i is represented as feature vector which is shown in Table 1.

Table 1 provides the information related to a word. It explains the particular words Document number, sentence number in which the word is present, the position of the word in a sentence, POS tag of the word and Class label for that particular word.

Class_Label attribute takes two possible values either YES or NO for the entity extraction classifier. YES means particular word is an entity or an instance of the respective class CLSNAME. If the word is not an entity or instance, it is labeled as NO. Position of word is calculated with respect to CLSNAME or its synonym word, i.e. CLSNAME Word is given position 0 and first left word from CLSNAME word is given position -1 and the position is decreased by -1 as we move left side of the sentence with respect to CLSNAME word. First right word from CLSNAME word is given position 1 and the position is increased by 1 as we move the right side of the sentence with respect to CLSNAME word.

Training data which is shown in Table 2 is discretized based on the sentence number of the word and position of the word within the sentence.

Here, a new attribute Region is introduced which is dependent on Sentence_Number and Position_Of_Word attribute which is described as follows:

Case 1: If Sentence_Number $< \alpha$ and Position_Of_Word $> -\beta$ and Position_Of_Word $< \gamma$ then Region is 'Nearer'.

Case2: If Sentence_Number $> \alpha$ and Position_Of_Word $< -\beta$ OR Position_Of_Word $> \gamma$ then Region is 'Far'.

The main objective of the introduction of Region attribute is, to define an initial section of the web page. We concentrated on the initial section of a web page because, the complete name of the product (including heading part) is generally placed in the initial section of the web page. The first occurrence of a complete entity name also comes under initial section of the web page. A region which comes under initial section is defined as Nearer otherwise the region is Far.

3.8.3. Labeling of training data

Those words which belong to an instance (entity) of product class name in the training data is labelled as either YES or NO. Criteria for labeling is as follows:

Case 1: If POS_Tag = NN or NNP and Region = Nearer then Class_Label is 'YES'.

Case2: If POS_Tag \neq NN or NNP and Region = Nearer or Far then Class_Label is 'NO'.

Here, we have considered a sufficient amount of features to label the word as an entity. Features such as POS_Tag of a words which are present before and after synonym word , distance of the words with respect to synonym word and Sentence number in which the word appears are taken into consideration.

3.8.4. Entity name extraction using J48 classifier

Decision tree classifier has been used for deciding the status of the words in the test documents. The decision tree is trained using training data generated from a variety of product classes. To decide the status of the words, words in the test documents are supplied to a classifier. The word which is classified as product class instance forms the entity name of the product class and its forms the row label in the spread sheet.

Table 1. Feature vector representation (entity).

Document_Id	Word	Sentence_Number	Position_Of_Word	POS_Tag	Class_Label
-------------	------	-----------------	------------------	---------	-------------

Table 2. Discretized feature vector representation (entity).

Document_Id	Word	POS_Tag	Region	Class_Label
-------------	------	---------	--------	-------------

3.9. Attribute-Value extraction

3.9.1. Classifier construction for attribute-value using J48 decision tree [Weka]

To decide words for attributes and their respective values, decision tree classifier is designed by generating feature vectors of each word in the sentences. Data is prepared for training and using decision tree, word is classified. For calculating the status of the word we have selected decision tree classifier over other types of classifier like Naïve Bayes [26], Neural Networks [27] and Support Vector Machines (SVM) [28] because of the following reasons:

- The feature vector designed for Entity and attribute-value extraction consists of both categorical and numerical attributes.
- The classification decision is sometimes impacted only by some set of attributes rather than all attributes, while in other types of classifiers always all the attributes have impacted classification decisions.
- Other classifiers depend on large data to estimate correct probability distributions, while the decision tree classifier uses nonparametric technique and can be trained even if data is limited.
- The other classifier performance depends on algorithm parameters like and probability distribution function (Naïve Bayes), activation function (NN) and kernel function (SVM).

The Java Implementation of a C4.5 algorithm in Weka tool is J48 Decision Tree Classifier. An extension of an ID3 algorithm is C4.5 algorithm [29]. An ID3 algorithm is one of the first algorithms for Decision tree implementation. The main reason for choosing C4.5 over ID3 is due to following reasons:

- The ID3 algorithm is not convenient for the In-

ternet Search activity [30] as it is oversensitive to features having a large number of values. In this situation, C4.5 is best suitable.

- The ID3 algorithm does not deal with numerical attributes. Our feature vector comprises both numeric and categorical attributes. In that case, C4.5 is the best suitable algorithm.
- Algorithm ID3 is not capable of handling missing values, while C4.5 is efficient and suitable for handling missing values.
- Post-pruning activity is very important to reduce error rates in unseen testing data. Algorithm C4.5 is capable of performing post pruning of decision trees. Algorithm ID3 does not support pruning activity.

Motivation of choosing J48 decision tree classifier over other classifiers has also been better explained in [25].

3.9.2. Building a features vector for each word

Using Tablet, Washing Machine and Tennis Racket product classes, training data is prepared by crawling related websites of different manufacturers or retailers. Every word in Document W_i is represented as a feature vector, which is shown in Table 3.

Table 3 provides the information related to a word. It explains the particular words Document number, sentence number in which the word is present, the position of the word in a sentence, POS tag of the word and Class label as WAV (Whether an Attribute or Value) for that particular word. WAV takes three possible values as A (if the word is an attribute), V (if the word is a value) and NAV (if the word is neither an attribute nor a value). Here, Document_Id, Sentence_Number, Position_Of_Word, POS_Tag are independent variables and WAV as the dependent variable.

Table 3. Feature vector representation for attribute-value pair.

Document_Id	Word	Sentence_Number	Position_Of_Word	POS_Tag	WAV
-------------	------	-----------------	------------------	---------	-----

Table 4. Discretized feature vector representation for attribute-value pair.

Document_Id	Word	Occurrence	WAV
-------------	------	------------	-----

Initially, the words which are having Alphanumeric (AN) data type or Numeric-Alpha (NA) data type are marked because these words are possible words for the value-measure pair word (VM). Then, Position of word is calculated with respect to these possible value words, that is the VM word is given position 0 and first left word from the VM word is given position -1 and the position is increased by -1 as we move the left side of the sentence with respect to VM word. First right from the VM word is given position 1 and the position is increased by 1 as we move the right side of the sentence with respect to VM word.

Above training data is discretized, which is depicted in Table 4.

Here, a new attribute Occurrence is introduced which is dependent on Position.Of.Word and POS.Tag attribute as follows:

Case 1: If POS.Tag=NN OR NNP and word is having the first occurrence either from the left side or right side with reference to VM word then Occurrence is set as 'First'.

Case2: If POS.Tag=NN or NNP and word is not having the first occurrence either from the left side or right side with reference to VM word then Occurrence is set as 'Later'.

Case3: For the rest of the words which dont fall under Case1 and Case 2, Occurrence is set as 'Random'.

Case4: For VM word Occurrence is set as 'FirstV'

3.9.3. Labeling of training data

Those words which belong to attribute name in the training data are labeled as 'A', words which belong to value word in the training data are labeled as 'V' and the words which doesnt belong to attribute or value word are labeled a 'NAV'. Criteria for labeling

is as follows:

An attribute WAV is dependent on Occurrence attribute as follows:

Case1: If Occurrence = First then the WAV is set as 'A' (Attribute).

Case2: If Occurrence = FirstV then the WAV is set as 'V' (Value).

Case3: If Occurrence=Later OR Occurrence=Random then the WAV is set as 'NAV' (Not an attribute or Value).

Here, we have considered a sufficient amount of features to label the word as attribute as well as value . Features such as an Occurrence of a word which considers distance of the words with respect to VM word and POS.Tag of the word are taken into account.

3.9.4. Attribute-value extraction using J48 classifier

For deciding the status of words in the test documents, decision tree classifier is used. The decision tree is trained using training data generated from a variety of product classes. Words in the test documents are supplied to the classifier to decide the status of the words. The words which are classified as an attribute (A) will form the columns of the spreadsheet for a corresponding entity name. The words which are classified as value words (V) will form the rows of the spreadsheet for corresponding attributes and entity name. The accuracy of the classifier for entity extraction and attribute-value extraction is measured using precision, recall and F-measure by N-Fold cross validation and the results are mentioned in the Section 5

4. Experimental setup

Experiments were performed on Windows Server 2012(R2) and softwares used are as follows:

- NetBeans IDE 8.0.2
- Java JDK 1.7.0.80
- Jsoup1.8.2 API is used, which is used to connect to the web document url and extracts all the sentences.
- Stanford POS (Part Of Speech) tagger v. 2.0 for tagging the terms of web pages.
- Word Net Stemmer for obtaining word stems.

5. Results and Discussions

Both the Classifiers for entity and attribute-value extraction are trained using class names as Tablet, Washing Machine and Tennis Racket. Effectiveness of algorithm which is described in Section 3 is evaluated by calculating classification accuracy (Cross validation) using N-Fold cross validation and 80-20 split i.e. 80 percent data are used for training and 20 percent data are used for testing.

5.0.5. Cross validation for entity extraction

Cross validation is performed on training as well as testing sets and results are described in Table 5 and Table 6 respectively.

Table 5. Accuracy of classifier built for entity name using 10-Fold Cross validation (On Complete Training Set).

10 Fold Cross validation	
Total Number of URLs Crawled	63
Total number of words extracted	29,999
Total number of words indicating the entity name	1490
Total number of words indicating not as entity name	28,509
Correctly classified Instances	29, 612 (98.71%)
Incorrectly classified Instances	387 (1.29%)
Weighted average Precision (for YES and,NO label)	98.7%
Weighted average Recall (for YES and,NO label)	98.7%
Weighted average F1 (for YES and,NO label)	98.7%

Table 6. Accuracy of classifier built for entity name using 80-20 split (80% data used for training and 20% data are used for testing).

80-20 split	
Total Number of URLs Crawled	63
Total number of words extracted	29999
Total number of words used for Training (80%)	23999
Total number of words used for Testing (20%)	6000
Correctly classified Test Instances (Either YES or NO)	5927 (98.78%)
Incorrectly classified Test Instances	73 (1.21%)
Weighted average Precision (for YES and,NO label)	98.8%
Weighted average Recall (for YES and,NO label)	98.8%
Weighted average F1 (for YES and,NO label)	98.8%

Total Number of URLs (web documents) crawled are 63. Table 5 indicates the accuracy of the model using N-Fold cross validation, where $N=10$. The total of number words extracted are 29,999 out of which 1490 words indicate the entity name and 28,509 words are not an entity. Using N-Fold cross validation (here, $N=10$), we obtain average precision, recall and F1 for both YES and NO label as 98.7%. A total of 29,6129 words are correctly classified into YES or NO category and 387 words are incorrectly classified.

Accuracy of classifier is also calculated using a 80-20 split technique as depicted in Table 6. In this method 80% of the total number of words is used for Training and 20% words are used testing. Out of 6000 words which are used for testing, 5927 words are correctly classified by the classifier into YES or NO category and 73 words are incorrectly classified. Using this method, we obtained average precision, recall and F1 for both YES and NO label as 98.8%.

5.0.6. Cross Validation for attribute-value pair extraction

Cross validation is performed on training as well as testing sets and results are described in Table 7 and Table 8 respectively.

Total Number of URLs (web documents) crawled are 70. Table 7 indicates the accuracy of the model using N-Fold cross validation, where $N=10$. The total of number words extracted are 186236 out of which 715 words indicate attribute, 476 words indicate value and 185032 words indicates neither as an attribute nor as value. We have extracted total number 12 distinct attribute-value pairs. Using N-Fold cross validation (here, $N=10$) we obtain precision, recall and F1 for class label 'A' as 100%, 99.3% and 99.7% and for the class label V as 99.2% for each precision, recall and F1 category. A total of 186223 words is correctly classified into A or V or NAV category and 13 words are incorrectly classified.

Accuracy of classifier is also calculated using a 80-20 split technique as depicted in Table 8. In this method 80% of the total number of words is used for Training and 20% words are used testing. Out of 37247 words which are used for testing, 130 words are correctly classified by the classifier as an attribute, 99 words are correctly classified as value, 37017 words are correctly classified as neither an attribute nor value. One word is incorrectly classified. Table 9 indicates the name of the attributes utilized for building the attribute-value extraction classifier.

Table 7. Accuracy of classifier built for value-measure pair using 10-Fold Cross validation (On Complete Training Set).

10 Fold Cross validation	
Total Number of URLs Crawled	70
Total number of words extracted	186236
Total number of words indicating attribute	715
Total number of words indicating value	476
Total number of words indicating not as an attribute or value	185032
Correctly classified Instances	186223 (99.99%)
Incorrectly classified Instances	13 (0.007%)
Precision for class label value as A	100%
Precision for class label value as V	99.2%
Recall for class label value as A	99.3%
Recall for class label value as V	99.2%
F-Measure for class label value as A	99.7%
F-Measure for class label value as V	99.2%

Table 8. Accuracy of classifier built for value-measure pair using 80-20 split (80% data used for training and 20% data are used for testing).

80-20 split	
Total Number of URLs Crawled	70
Total number of words extracted	186236
Total number of words used for Training (80%)	148989
Total number of words used for Testing (20%)	37247
Total number of words indicating attribute	130
Total number of words indicating value	99
Total number of words indicating not as an attribute as well as value	37017
Total number of distinct attribute-value pairs	12
Precision for class label value as A	100%
Precision for class label value as V	100%
Recall for class label value as A	100%
Recall for class label value as V	99.0%
F-Measure for class label value as A	100%
F-Measure for class label value as V	99.5%

Table 9. Attributes used for building the classifier.

Sr.No	Class	Attribute Names
1.	Tablet	Touchscreen, Camera, RAM, Processor
2.	Washing Machine	Consumption, Capacity, Weight, Speed
3.	Tennis Racket	Size, Weight, Pattern, Balance

Generation of spreadsheet requires building two classifiers, one for identifying entities and other for identifying attribute-value pairs. This task is very challenging because it should work for general products.

F1-measure is basically a harmonic mean of Precision and Recall. The high value of F1-measure indicates more accuracy. A classifier should be designed using a known product with a known attribute, but it has to be applied for unknown products and their attributes for practical purpose. The classifier is trained using 'n' percent of known data and 'm' percent of test data by randomly selecting from known data. A low value of 'n' may lower accuracy of the classifier and high value of 'n' may increase the accuracy for the known products but it may introduce bias for the known products as shown in Table 10 and Table 13. A high value of 'n' may

reduce generalization in classifier and may lower accuracy for the unknown product so experimentation is done to decide the value of 'n' based on performance measures of unknown products and results are given in Table 12 and Table 15. Table 10 depicts Weighted F1-measure for the various percentages of "n-m" splits for the entity name classifier. Table 11 describes confusion matrix for the various percentages of "n-m" splits. Here, total labeled data consist of 29,999 words. Table 12 depicts the performance (Prediction Accuracy) of the entity classifier for various percentages of "n-m" splits when the classifier has been applied for the 59 unknown instances of "Mobile Phone" class. As seen from Table 12, we obtained the highest accuracy for n=80. Fine tuning can be done to decide the exact value of 'n' arrived, but practically it may not be required if accuracy is not enhancing at a very high rate.

Table 10. Accuracy of Classifier built for entity name using "n-m" split.

Weighted Average F1-measure (Accuracy)				
70-30 split	75-25 split	80-20 split	85-15 split	90-10 split
98.4%	98.6%	98.8%	98.7%	97.9%

Table 11. Confusion Matrix for the entity name classifier using "n-m" split.

Confusion Matrix										
	70-30 split		75-25 split		80-20 split		85-15 split		90-10 split	
	Pr ^b (YES)	Pr ^b (NO)	Pr ^b (YES)	Pr ^b (NO)	Pr ^b (YES)	Pr ^b (NO)	Pr ^b (YES)	Pr ^b (NO)	Pr ^b (YES)	Pr ^b (NO)
Ac ^a (YES)	1623	97	1375	75	1109	48	833	32	545	33
Ac ^a (NO)	44	7236	31	6019	25	4818	20	3615	25	2397

^a Actual Class

^b Predicted Class

Table 12. Accuracy of entity classifier for unknown dataset.

Average Prediction Accuracy (Both YES and NO).				
70-30 split	75-25 split	80-20 split	85-15 split	90-10 split
93.4%	93.7%	94.5%	94.1%	94.0%

Table 13. Accuracy of Classifier built for attribute-value using "n-m" split.

Weighted Average F1-measure (Accuracy)				
70-30 split	75-25 split	80-20 split	85-15 split	90-10 split
98.6%	98.9%	100%	98.9%	98.8%

Same strategy as discussed above has been applied to divide the labeled data (of value measure pair) into "n-m" split for attribute-values. Table 13 depicts Weighted F1-measure for the various percentages of "n-m" splits for an attribute-value classifier. Table 14 describes confusion matrix for the same. Here, total labeled data consist of 18,6236 words. Table 15 describes the performance (Prediction Accuracy) of the attribute-value classifier for various percentages of "n-m" splits when the classifier has been applied for the 59 unknown instances of "Television" class. Here, labeled data consist of 18,6236 words. As seen from Table 15, we obtained the highest accuracy for n=80. We can draw a con-

clusion from Table 12 and Table 15 that, if 'n' is very less then classification generalization is not enough captured resulting in less accuracy. If 'n' is very high, then overfitting may occur in classification and bias is introduced for the specific product and the classifier may not give enough accuracy for the general unknown product so, from the experimentation, it is suggested to have 80-20 split for getting good accuracy.

5.1. Evaluation on different product classes

The proposed technique is used to generate a spreadsheet of product classes of different categories. Ac-

Table 14. Confusion Matrix for the attribute-value classifier using "n-m" split.

Confusion Matrix															
	70-30 split			75-25 split			80-20 split			85-15 split			90-10 split		
	Pr ^b (NAV)	Pr ^b (V)	Pr ^b (A)	Pr ^b (NAV)	Pr ^b (V)	Pr ^b (A)	Pr ^b (NAV)	Pr ^b (V)	Pr ^b (A)	Pr ^b (NAV)	Pr ^b (V)	Pr ^b (A)	Pr ^b (NAV)	Pr ^b (V)	Pr ^b (A)
Ac ^a (NAV)	5519	2	2	46267	0	1	37017	0	0	27766	2	3	18501	4	6
Ac ^a (V)	2	139	1	1	122	0	1	99	0	1	76	1	5	35	5
Ac (A)	2	2	202	0	1	166	0	0	130	1	1	84	2	3	57

^a Actual Class

^b Predicted Class

Table 15. Accuracy of attribute-value pair classifier for unknown dataset.

Average Prediction Accuracy (For 'A', 'V' and 'NAV' class)				
70-30 split	75-25 split	80-20 split	85-15 split	90-10 split
94.8%	95.1%	95.5%	95.2%	94.3%

curacy is evaluated using precision, recall and F1 measure. The results of the experimentation of product classes such as Mobile Phone, Mixer Grinder, Television and Electric Timer are described in the following sections:

5.1.1. Product class name: Mobile-Phone

Input: Search query specifying class name CLSNAME as Mobile-Phone, attribute as Price, maximum number of URLs crawled (n) to limit the search space as 20.

Output: Spreadsheet WSP consisting of Price, Camera, RAM, Processor and Touchscreen as columns and rows of a spreadsheet as their respective values. Sample Spreadsheet WSP for Mobile-Phone is shown in Table 16.

As mentioned in Table 17, for a particular attribute, "Total Relevant Doc" column corresponds to the total number of relevant documents having product class name as Mobile Phone and the attribute word. "Relevant Docs Retrieved" indicates actual relevant documents retrieved by the algorithm for a particular attribute or entity name. "Total Doc

Retrieved" corresponds to the total number of documents retrieved for an attribute or entity name. As mentioned in Table 17, we obtained 312 "Relevant Docs" Retrieved, 332 "Total Doc Retrieved", precision as 0.939, "Total Relevant Doc" as 331, Recall as 0.942 and F1 as 0.945 for entity name. For camera attribute, we obtained 160 "Relevant Docs Retrieved", 195 "Total Doc Retrieved", precision as 0.820, "Total Relevant Doc" as 200, Recall as 0.80 and F1 as 0.809.

Precision, Recall and F1 is calculated as follows:

$$Precision = \frac{\sum \text{Relevant Doc's Retrieved}}{\sum \text{Total Doc Retrieved}} \quad (1)$$

$$Recall = \frac{\sum \text{Relevant Doc's Retrieved}}{\sum \text{Total Relevant Doc's}} \quad (2)$$

$$F1 = \frac{2 * \text{Average Precision} * \text{Average Recall}}{\text{Average Precision} + \text{Average Recall}} \quad (3)$$

Table 16. Spreadsheet for Mobile-Phone.

Entity-name	Price	Camera	RAM	Processor	Touchscreen
Apple iPhone6	29,999	8MP	1GB	1.84Ghz	4.7inches
CoolPad Note5	10,786	13 MP	4GB	1.5 GHz	5.5 inch
OnePlus 3T	29,999	16MP	64GB	2.35 GHz quadcore	5.5 inch
Vivo V5	17,499	13MP, 20MP	4GB	M10 Motion Co-processor	5.5 inch

Table 17. Precision, Recall and F1 calculation for Mobile-Phone class.

Product Name: Mobile-Phone						
Attribute /Instance Name	Relevant Docs Retrieved	Total Doc Retrieved	Precision	Total Relevant Doc	Recall	F1
Entity-Name	312	332	0.939	331	0.942	0.945
Camera	160	195	0.820	200	0.80	0.809
RAM	152	179	0.8498	318	0.477	0.610
Processor	152	179	0.849	197	0.771	0.808
Touchscreen	112	140	0.80	151	0.741	0.769
Total	866	1016		1064		
Average			0.852		0.813	0.823

5.1.2. Product class name: Mixer Grinder

Input: Search query specifying class name CLSNAME as Mixer Grinder, attribute as Price, maximum number of URLs crawled (n) to limit the search space as 20.

Output: Spreadsheet WSP consisting of Price, Warranty, Capacity, Weight and Height as columns and rows of a spreadsheet as their respective values. Sample Spreadsheet WSP for Mixer Grinder is shown in Table 18.

As mentioned in Table 19, we obtained 350 "Relevant Docs Retrieved", 381 "Total Doc Retrieved", precision as 0.918, "Total Relevant Doc" as 385, Recall as 0.909 and F1 as 0.913 for entity name. For warranty attribute, we obtained 210 "Relevant Docs Retrieved", 230 "Total Doc Retrieved", precision as 0.869, "Total Relevant Doc" as 245, Recall as 0.857 and F1 as 0.862.

5.1.3. Product class name: Television

Input: Search query specifying the class name CLSNAME as Television attribute as Price, maximum number of URLs crawled (n) to limit the search space as 20.

Output: Spreadsheet WSP consisting of Price, Television Size, Response Time, Weight and Angle as columns and rows of a spreadsheet as their respective values. Sample Spreadsheet WSP for Television is shown in Table 20.

As mentioned in Table 21, we obtained 209 "Relevant Docs Retrieved", 225 "Total Doc Retrieved", precision as 0.928, "Total Relevant Doc" as 237, Recall as 0.881 and F1 as 0.903 for entity name. For "Response Time" attribute, we obtained 80 "Relevant Docs Retrieved", 101 "Total Doc Retrieved", precision as 0.792, "Total Relevant Doc" as 110, Recall as 0.727 and F1 as 0.758.

5.1.4. Product class name: Electric Timer

Input: Search query specifying class name CLNAME as Electric Timer, attribute as Price, maximum number of URLs crawled (n) to limit the search space as 20.

Output: Spreadsheet WSP consisting of Timer size, Weight, Humidity and Current as columns and rows of a spreadsheet as their respective values. Sample Spreadsheet WSP for Electric Timer is shown in Table 22.

As mentioned in Table 23, we obtained 70 "Relevant Docs Retrieved", 74 "Total Doc Retrieved", precision as 0.945, "Total Relevant Doc" as 80, Recall as 0.875 and F1 as 0.908 for entity name. For Humidity attribute, we obtained 50 "Relevant Docs" Retrieved, 60 "Total Doc Retrieved", precision as 0.833, "Total Relevant Doc" as 65, Recall as 0.769 and F1 as 0.799. Table 24 and Table 25 shows the performance comparison of the proposed method with some of the prominent entity extraction and attribute-value extraction methods discussed in the literature. These methods have evaluated their performance on limited web corpus or a single site. We have evaluated the performance of the proposed method by crawl-

ing over almost every manufacturer or retailer websites.

Our work is different than the work described in [22–24] in such a way that, in the work which is described in this paper we automatically extract attribute-value pairs. We perform the extraction process on different product description pages of various retailers or manufacturers. As compared to online reviews these pages are highly unstructured. The appearance of the attribute name is different on the product description page as compared with online reviews. For example, in the product description page of any manufacturer, RAM is mentioned as an attribute. While in online reviews, customer will mention memory capacity as an attribute.

Table 18. Spreadsheet for Mixer-Grinder.

Entity-name	Price	Warranty	Capacity	Weight	Height
Kenstar	1,499	1 year	1.25 L	6.6kg	16.6inch
Philips HL1632	3,099	2 years	0.52 l	7.2 kg	43.2 cm
Prestige stylo	2,999	2 Years Manufacturer	0.5 liters	2 kg	50.1 cm
Pigeon Gusto	1,399	1 Year Company	0.6 L	3.7 kg	30.99 cm

Table 19. Precision, Recall and F1 calculation for Mixer-Grinder class.

Product Name: Mixer-Grinder						
Attribute /Instance Name	Relevant Docs Retrieved	Total Doc Retrieved	Precision	Total Relevant Doc	Recall	F1
Entity-Name	350	381	0.918	385	0.909	0.913
Warranty	210	230	0.869	245	0.857	0.862
Capacity	50	70	0.714	86	0.813	0.759
Weight	85	105	0.809	115	0.739	0.772
Height	70	80	0.875	89	0.786	0.828
Total	765	866		920		
Average			0.883		0.831	0.856

Table 20. Spreadsheet for Television.

Entity-name	Price	Television size	Response Time	Weight	Angle
Micromax	15,786	32inches	9 milliseconds	6 kg	172degrees
Daiwa	11,666	19.5 inch	11ms	2.8 kg	178-degree
Haier	12,877	102.6 cm	12milliseconds	8.5 kg	176-degree
Philips	18,495	85 cm	13ms	6.5 kg	176 degree

Table 21. Precision, Recall and F1 calculation for Television class.

Product Name: Television						
Attribute /Instance Name	Relevant Docs Retrieved	Total Doc Retrieved	Precision	Total Relevant Doc	Recall	F1
Entity-Name	209	225	0.928	237	0.881	0.903
Television size	147	165	0.890	181	0.812	0.849
Response Time	80	101	0.792	110	0.727	0.758
Weight	71	90	0.788	92	0.771	0.779
Angle	53	60	0.883	62	0.854	0.868
Total	560	641		682		
Average			0.873		0.821	0.846

Table 22. Spreadsheet for Electric Timer.

Entity-name	Price	Timer size	Weight	Humidity	Current
GIC Micon	1,598	18.8 cm	52 grams	4587rh	85mah
Selec	\$23.37	15.6 cm	47 g	4569rh	77mah
L& T Micon	\$26.37	19.6 cm	57 grams	3890rh	78mah
Kaycee	\$25.97	18.6 cm	49 grams	4440rh	79mah

Table 23. Precision, Recall and F1 calculation for Electric Timer class.

Product Name: Electric Timer						
Attribute /Instance Name	Relevant Docs Retrieved	Total Doc Retrieved	Precision	Total Relevant Doc	Recall	F1
Entity-Name	70	74	0.945	80	0.875	0.908
Timer size	65	80	0.812	84	0.773	0.792
Weight	76	88	0.863	90	0.844	0.853
Humidity	50	60	0.833	65	0.769	0.799
Current	40	50	0.800	55	0.727	0.761
Total	261	352		374		
Average			0.741		0.697	0.718

Table 24. Comparison of proposed entity extraction method with the performance of methods in the literature.

Reference	Class	Performance Parameter	Dataset
[8]	City, Film , Scientist	Precision (52%, 72%, 64% respectively)	Limited Web Corpus
[15]	LOC, MISC,ORG, PER	F1-Measure (87.44%, 72.87%, 78.92%, 90.51% respectively)	CONLL-2003 English Test Set
[16]	Location, Organization, Person	Precision (70%, 85 %, 88% respectively)	CONLL-2003
[17]	Artifact, Organization, Location, Person	F1- Measure (50.16%, 80.44%, 88.57%, 87.81% respectively)	CRL NE
Our method	Mobile Phone, Mixer Grinder, Television, Electric Timer	Precision (93.9%, 91.8%, 92.8%, 94.5% respectively) F1-Measure (94.5%, 91.3%, 90.3%, 90.8% respectively)	All related web pages returned by search engine.

Table 25. Comparison of proposed value-measure extraction method with the performance of methods in the literature.

Reference	Category	Performance Parameter	Dataset
[19]	Football & Tennis	Recall (Using Co-EM) 55.48% & 75.91% respectively	Single Site
[20]	Tweets, Web-Links, Tweets + Web-Links	F1-measure (51.6%, 42.9% & 59.5% respectively)	Only Tweets, Only Web-Links, Tweets + Web-Links
[21]	Wine and Shampoo	F1-measure (58.15%), Precision (57.05%), Recall (59.66%)	Single Site
Our method	Mobile Phone, Mixer Grinder, Television, Electric Timer	F1-Measure (78.1%, 81.3%, 81.4%, 66.7% respectively) Recall (75.5%, 77.5%, 78.8%, 64.9% respectively)	All related web pages returned by search engine

6. Conclusions and Future work

A huge amount of product related data is spread over various web pages. Every manufacturer or retailer displays attribute (feature) information related to a particular class of product, but this information is insufficient. The user has to crawl different websites for choosing the right product because different values for same feature is available on various websites. If any manufacturer introduces new feature for a particular product, it is updated manually on their website. In this work, for a particular class of products, exceptional product differentiation based on features has been carried out by introducing novel concept of web-spreadsheet which is a solution to the above discussed problem.

The web-spreadsheet is generated dynamically and it stores consolidated information of product names and their respective attribute-value pairs, for a particular class of product. We have crawled nearly every websites of retailers for generating spreadsheet. The user will create search query which consists of a particular class of product for which the spreadsheet has to be generated. The search query is fired to search engine which returns related URLs of nearly every product manufacturers or retailers. Web documents corresponding to these URLs are crawled one by one and product names and attribute-value pairs are automatically extracted from unstructured natural language documents.

Using separate J48 decision tree based classifier, entities (Product names) and attribute-value pairs are extracted successfully. Decision tree classifier is trained by using product classes of different categories. As training data are very expensive to generate, we experimentally showed that, the trained classifier can be used to test (classify) product classes without requiring training data of each class. In this work we have dealt with Alphanumeric and Numeric-Alpha type value-measure pair word. We have extensively performed experiments on various product classes and we obtained encouraging results in each product category. The proposed method accuracy is exceptionally good as compared with many entity and attribute-value extraction techniques as shown in Table 24 and Table 25. Immediate Future work is the classification accuracy can be improved

by utilizing more detailed information from ontologies and using word phrase information at statement and document level. Similar approach as we have discussed in this paper can also be adopted for different kinds of problems such as comparing services. Various classifiers can be designed and tested for the same.

Acknowledgment

Authors would like to thank Computer Science and Engineering Department, Visvesvaraya National Institute of Technology, Nagpur for providing necessary facilities for carrying out this research work.

References

1. A. Kopliku, K. Pinel-Sauvagnat and M. Boughanem, Aggregated Search: A New Information Retrieval Paradigm, *ACM Computing Surveys (CSUR)* **46** (3) (2014) 41.
2. J. Turmo, A. Ageno and N. Catal, Adaptive information extraction, *ACM Computing Surveys (CSUR)* **38** (2) (2006) 4.
3. O. Etzioni, A. Fader, J. Christensen, S. Soderland and M. Mausam, Open Information Extraction: The Second Generation, in *Proc. of the International Joint Conference on Artificial Intelligence*, (IJCAI 11), (2011), pp. 3-10.
4. J. Wang, C. Chen, C. Wang, J. Pei, J. Bu, Z. Guan and W.V. Zhang, Can we learn a template-independent wrapper for news article extraction from a single training site?, in *Proc. of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM(Paris, 2009), pp. 1345-1354.
5. S. Liu, Experiences with and reflections on text summarization tools, *International Journal of Computational Intelligence Systems* **2** (3) (2009) 202-218.
6. C.H. Chang, M. Kaye, M. R. Girgis and K. F. Shaalan, A survey of web information extraction systems, *IEEE transactions on knowledge and data engineering* **18** (10) (2006) 1411-1428.
7. X. Wu, T. W. Ling, M.L. Lee and G. Dobbie, Designing semistructured databases using ORA-SS model, in *Proc. of the 2nd International Conference on Web Information Systems Engineering (WISE)*, IEEE Computer Society (Japan,2001), pp. 171-180.
8. O. Etzioni, M. Cafarella, D. Downey, A. M. Popescu, T. Shaked, S. Soderland, D. S. Weld and A. Yates, Unsupervised named-entity extraction from the web: An experimental study, *Artificial intelligence* **165** (1) (2005) 91-134.

9. D. Rosaci, CILIOS: Connectionist Inductive Learning and Inter-Ontology Similarities for Recommending Information Agents, *Information Systems* **32**(6) (2007) 793-825.
10. H. Alani, S. Kim, D. E. Millard, M. J. Weal, W. Hall, P. H. Lewis, and N. R. Shadbolt, Automatic ontology-based knowledge extraction from web documents, *IEEE Intelligent Systems*, **18**(1) (2003) 14-21.
11. D. Rosaci, Finding semantic associations in hierarchically structured groups of Web data, *Formal Aspects of Computing (FAOC)* **27**(5) (2015) 867-884. DOI: 10.1007/s00165-015-0337-z. 2015. Springer.
12. D. Rosaci, G. M. Sarn and S. Garruzzo, MUADDIB: A distributed recommender system supporting device adaptivity, *ACM Transactions on Information Systems (TOIS)*, **27**(4) (2009) 1-41.
13. MUADDIB Project URL (2009). <http://www.muad.altervista.org>.
14. B. D. Jesudas and B. Gurumoorthy, Domain Specific Named Entity Extraction for Modeling and Populating Ontologies, in *Proc. International Conference on Research into Design*, Springer (Singapore, 2017), pp. 751-760.
15. A. McCallum and W. Li, Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons, in *Proc. of the seventh Association for Computational Linguistics conference on Natural language learning*, Association for Computational Linguistics (2003), pp. 188-191.
16. P. P. Talukdar, T. Brants, M. Liberman and F. Pereira, A context pattern induction method for named entity extraction, in *Proceedings of the 10th Conference on Computational Natural Language Learning (CoNLL-X)*, Association for Computational Linguistics (New York, 2006), pp. 141-148.
17. M. Asahara and Y. Matsumoto, Japanese named entity extraction with redundant morphological analysis, in *Proc. of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, (2003), pp. 8-15.
18. J. B. Antony and G. S. Mahalakshmi, Content-based information retrieval by named entity recognition and verb semantic role labelling, *Journal of universal computer science* **21** (3) (2015) 1830-1848.
19. R. Ghani, K. Probst, Y. Liu, M. Krema and A. Fano, Text mining for product attribute extraction, *ACM SIGKDD Explorations Newsletter* **8** (1) (2006) 41-48.
20. S. Panem, M. Gupta and V. Varma, Structured information extraction from natural disaster events on twitter, in *Proc of the 5th International Workshop on Web-scale Knowledge Representation Retrieval & Reasoning*, ACM (2014), pp. 1-8.
21. K. Shinzato and S. Sekine, Unsupervised Extraction of Attributes and Their Values from Product Description, in *Proc. Sixth International Joint Conference on Natural Language Processing (IJCNLP)*, (2013) , pp. 1339-1347.
22. J. Jin, P. Ji and R. Gu, Identifying comparative customer requirements from product online, *Engineering Applications of Artificial Intelligence* **49** (2016) 61-73.
23. A. M. Popescu and O. Etzioni, Extracting product features and opinions from reviews, in *Proceedings of the Human Language Technology Conference and the Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP)*, Springer (London, 2007), pp. 9-28.
24. M. Hu and B. Liu, Mining and summarizing customer reviews, in *Proc. of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, ACM (Aug 2004), pp. 168-177.
25. A. R. Thakare and P. S. Deshpande, Comparative Search of Entities, *International Journal of Software Engineering and Knowledge Engineering* **27** (8) (2017) 1333-1357. DOI: 10.1142/S0218194017500498.
26. Y. Yang and X. Liu, A re-examination of text categorization methods, in *Proc. of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, (1999), pp. 42-49.
27. F. Sebastiani, Machine learning in automated text categorization, *ACM computing surveys (CSUR)* **34** (1) (2002) 1-47.
28. T. Joachims, Text categorization with support vector machines: learning with many relevant features, in *Proc. of ECML-98, 10th European Conference on Machine Learning*, (Germany, 1998), pp. 137-142.
29. J. R. Quinlan, C4. 5: Programs for Machine Learning (Elsevier, Netherlands, 2014).
30. B. Hssina, A. Merbouha, H. Ezzikouri and M. Erritali, A comparative study of decision tree ID3 and C4. 5, *International Journal of Advanced Computer Science and Applications* **4** (2) (2014) 13-19.