Web-based Extraction of Technical Features of Products

Sebastian Schmidt, Herbert Stoyan

Chair of Artificial Intelligence
Friedrich-Alexander-Universität Erlangen-Nürnberg
Am Weichselgarten 9
91058 Erlangen
sebastian.schmidt@informatik.uni-erlangen.de
stoyan@informatik.uni-erlangen.de

Abstract: We present a novel symbolic approach to extract domain-specific technical features of products from large German corpora. Our prototypical implementation extracts terms like "Auflösung" (resolution), "Speicherplatz" (storage capacity), etc. The proposed methods depend on manually added lists of technical measures of the target domain (in our case measures like "Megapixel" or "MB").We applied the extraction in the domain of digital cameras on the internet using Google.

1 Introduction

When one wants to buy a technical product like a digital camera and is not an expert in the field of optical technology, one usually consults the internet to gain knowledge about the market situation (producers, vendors, products) and the state of the art. We typically have a specific application for our new investment in mind. While browsing, we learn about different product classes, their advantages and disadvantages. When sure about a product class to target, a careful person wants to know as many parameters as possible of products from this class. As of digital cameras, these might be features like weight, resolution of the CCD-Sensor, storage capacity and others. After having extracted (and understood) all the technical features, our hypothetical buyer then should extract all the values of those features for each product available, to assemble a comparison chart to help him making an informed decision. To help people with this time consuming process is our ultimate goal. As described above, we see the following steps when searching for technical products on the internet for a specific need:

(1) Extracting product-features for the desired product-class to generate a template for concrete offers.

(2) Extracting concrete offers and filling this template.

(3) Mapping the desired application to constraints of technical features and suggesting a set of adequate products.

This paper deals only with the automatic extraction of technical features step (1), while step (2) and (3) will be adressed in our future research. Automatic extraction of features has the advantage, that it is useful in dynamic scenarios as our task of finding product information. Products (and accordingly the websites describing them) change a lot. A handcrafted list of features will be outdated very soon, because technical innovations emerge on a regular basis.

2 Related Work

Our work is related to several subfields of Textmining. The next few paragraphs list the most relevant approaches and compare our ideas to the state of the art.

2.1 Terminology Finding

Terminology Finding deals with the problem of identifiying relevant terms in a domain-specific corpus of documents. There are basically two approaches: symbolic approaches that rely on syntactic characterisitics of terms, namely noun phrases, and statistical approaches that exploit the fact that the words composing a term tend to be found close to each other and reoccuring (collocations) [JB01], [CH90]. Using noun phrases tends to produce too many non-terms (low precision), while using collocations misses many low frequency terms, terms with variations, and terms with only one word. Because we analyse German websites, and German is a language of very many compounds, collocations tend to occur less frequent in German text than, for example, English text.

2.2 Template Filling

Template Filling is a subtask of Information Extraction (IE). It addresses the problem of extracting strings directly from a document to fill slots of a specified template. Template Filling is used in a variety of tasks, e.g. extracting job postings in newsgroups [CM98], seminar announcements, restaurant guides, or apartment rental advertisements. At first glance, our approach of extracting technical features in step (1) is not related to Template Filling. But when considering step (2), one sees that the extracted features describe templates to be filled later on. Current IE-Systems are not able to generate situation-dependant templates, that may change over time.

2.3 Opinion Mining

Opinion Mining aims at extracting users' opinions about products and suchlike. Users often express their general opinion by giving "subopinions" of different aspects about the entity to be judged. Hu and Liu built a system to mine user opinions about digital cameras [HL04]. They found, that technical features of products are often referenced in customer reviews. In subjective texts, these features co-occur with adjectives that express opinions, e.g. "The picture quality is lousy." By extracting candidate features that are noun phrases, they prune the set of candidates by applying the rule of adjective co-occurring. Because our application aims more at objective texts, we can not make use of adjectives, but our heuristic of identifying features is similar, as will be shown later on.

3 Our Approach

Our approach bases on one main heuristic. While analysing websites on digital cameras (or technical products as a whole), we found that most of the technical features are quantified by measured values. The feature "resolution" of a digital camera is often measured in "Megapixel", "MP", "millions of pixels" and suchlike. To find good candidates for technical features means to find patterns of the form

(1) <NOUN> … <NUMBER> … <MEASURE>, and

(2) <NUMBER> … <MEASURE> … <NOUN>,

where "…" denotes arbitrary tokens, that are not nouns, numbers, or measures. Measures are strings that where manually assigned by us to the class of measures. Therefore we created a list of 39 measures that are common in websites about digital cameras. Such measures include strings about weight (kg, kilogram, g, …), dimension (cm, Zentimeter, …), optical aspects (Megapixel, dpi, ", …), time (h, min, Sekunden, …), and storage (Megabyte, MB, …). Figure 1 gives an overview of the architecture of our Java implementation.

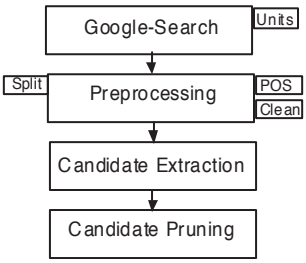

Figure 1: Architecture

With our list of measures and the two terms "Digitalkamera" (digital camera) and "technische Eigenschaften" (technical properties) we generate pairs of Google queries (search term and a unit) and evaluate the first 200 listed results for each query. Thus, we download about 8000 web pages to gain statistical significance. Each webpage is first pre-processed, before candidates of technical features are extracted. To find the patterns, all the HTML and JavaScript content of the website must be removed. We use the open-source library htmlparser [Du05] to accomplish this task. Due to the unavailability of a German part-of-speech tagger written in java, we developed simple heuristics to tag nouns and to do the sentence splitting. Usually sentences end with punctuation marks (":", ";", ".", "!", "?"). The dots "." play an important role when detecting sentence boundaries, because they may also occur in

abbreviations (e.g. “ca.”) and numbers (e.g. “1.50”). To handle abbreviations we created a lexicon of common German abbreviations, dots in numbers can be easily checked.

After the website is split into sentences, we tag nouns and cardinalities with the following heuristic:

(3) A token is a noun, if it is not a stop word, not an abbreviation, is longer than three and starts with an upper case letter followed only by lower case letters.

(4) A cardinality is a token that only contains numbers or dots, while having more than one digit.

After pre-processing, we apply a search for the patterns (1) and (2) in our downloaded corpus and associate each noun that matched one of the patterns to the corresponding unit. Each pair of unit and noun has a counter attached that is increased whenever another occurrence is found. The extracted candidates are finally pruned by two criteria: First a candidate must occur at least 20 times to be considered a feature. Our experiments have shown that 20 occurrences of a candidate to be a feature seems to be a reasonable value. If more (or less) than 200 websites or returned from Google, this value must be adjusted accordingly. Second we pass that candidate through a morphological tagger for German described [Ha94]. This tool does a morphological analyse of the word and outputs all categories (verb, noun, adjective, …) the word can be. Only if the tagger lists the possibility of being a noun, the candidate is rated as a feature. The morphological analysis is necessary to improve our somehow crude heuristic to identify German nouns. By applying it on the candidates, we accomplish higher computational efficiency, because applying the analysis on each token of the roughly 8000 websites would be too time consuming.

4 Results

Figure 2 lists all the technical features (along with their counts) we extracted with our approach. Technical measures are listed in the columns, the extracted terms are below each measure together with their counts.

gb		**Gramm**		**g**		**Megabyte**	
Festplatte	36	Fliegengewichte	40	Caplio	22	Speicher	34
Bundle	38	Gewicht	44	Gewicht	54	**Minute**	
Canon	26	**Megapixel**		**Gigabyte**		Akkus	32
Microdrive	39	Zoom	26	Speicherkarte	21	**Zoll**	
kg		Sensor	22	**pixel**		Anzeige	30
Gewicht	24	Hersteller	44	Auflösung	70	Display	30
Kilogramm		Auflösung	58	Bildwandler	50	**MB**	
Gewicht	24	**mm**		**Sekunden**		Karte	47
iso		Brennweite	22	Auslöseverzögerung	22	Festplattenspeicher	26
Zoom	33	**cm**		**lw**		Speicher	80
mp		Mindestabstand	32	Belichtungskorrektur	24	Speicherkapazität	22
Auflösung	54	Länge	22	Schritten	22	Speicherkarte	30

Tabelle 1: Extrahierte technische Eigenschaften und deren Häufigkeiten

5 Discussion

When looking at the results, we think our approach has potential. Nevertheless, to accomplish better precision we have to improve our algorithms. We see the following weaknesses:

Non-metric features: By using patterns that involve nouns and measures, only metric quantified features can be found. Nominal ones, like the colour of the casing of a product, can not be found.

Structural pre-processing: By simply stripping the websites off their HTML Code, we introduce a large amount of noise to our corpus. If the websites could be structurally analysed to identify navigation bars, forum headers and so on, we could eliminate all those parts, that hardly bear any content. The precision of the extracted features would rise, as was investigated by us. Many of the wrongly extracted features are not part of the relevant content of the website, they occur in link sections and suchlike.

Classification of correct Sentences: Another approach to filter unwanted noise is to classify the sentences for grammatical correctness in our corpus. Because we only do sentence boundary detection and splitting, many of those noisy parts end up in sentences, that are in no way correct German sentences (e.g. paragraph headers). Our department currently works on the problem to decide, whether or not a given sentence is a German utterance.

Need of technical units: at the moment, one has to assemble a list of common technical measures for the domain to extract in. This list has to be handcrafted once, as long as no new technical measures are introduced in the domain. We will try to extend our algorithms to find technical units as well. It is not known if that can be accomplished automatically for the application to bootstrap itself. If not, one idea is to start with a partial list of measures and trying to extend it.

References

[CH90] Church, K.; Hanks, P.: Word Association Norms, Mutual Information and Lexicography. Computational Lingustitics Volume 16, No. 1, MIT Press, Cambridge, 1990: S 22-29.

[CM98] Califf, M. E.; Mooney, R. J.: Relational Learning of Pattern-Match Rules for Information Extraction. Working Notes of AAAI Spring Symposium on Applying Machine Learning to Discourse Processing, AAAI Press, Menlo Park, 1998; S. 6-11.

[Da96] Daille, B: Study and Implementation of Combined Techniques for Automatic Extraction of Terminology. The Balancing Act: Combining Symbolic and Statistical Approaches to Language. MIT Press, Cambridge, 1996

[Du05] Udani, D.: The HTML Parser. http://htmlparser.sourceforge.net. 2005.

[Ha94] Hanrieder, G.: MORPH. In (Hausser, R. Hrsg.): Linguistische Verifikation. Dokumentation zur ersten Morpholympics 1994. Niemeyer, Tübingen, 1996: S. 53-66.

[HL04] Hu, M.; Liu, B.: Mining and Summarizing Customer Reviews. In (Kohavi, R., Hrsg.): Proceedings of the 2004 ACM SIGKDD international conference on Knowledge discovery and data mining, Seattle, 2004. ACM[JB01] Jacquemin, C.; Bourigault, D.: Term extraction and automatic indexing. In (Mitkov, R. Hrsg.): Handbook of Computational Linguistics. Oxford University Press, 2001.