Originally published in: Proceedings of the WikiAI 09 - IJCAI Workshop:
User Contributed Knowledge and Artificial Intelligence: An Evolving Synergy. 2009. Pp. 3-8.

LRT$_{wiki}$: Enriching the Likelihood Ratio Test with Encyclopedic Information for the Extraction of Relevant Terms

Niklas Jakob and **Mark-Christoph Müller** and **Iryna Gurevych**
Ubiquitous Knowledge Processing Lab
Technische Universität Darmstadt
Hochschulstr. 10, 64289 Darmstadt, Germany
{njakob, chmark, gurevych}@tk.informatik.tu-darmstadt.de

Abstract

This paper introduces LRT$_{wiki}$, an improved variant of the Likelihood Ratio Test (LRT). The central idea of LRT$_{wiki}$ is to employ a comprehensive domain specific knowledge source as additional "on-topic" data sets, and to modify the calculation of the LRT algorithm to take advantage of this new information. The knowledge source is created on the basis of Wikipedia articles. We evaluate on the two related tasks product feature extraction and keyphrase extraction, and find LRT$_{wiki}$ to yield a significant improvement over the original LRT in both tasks.

1 Introduction

The identification of the most relevant terms[1] in a document collection is one of the pervasive tasks in natural language processing. A fundamental application of this task is keyphrase extraction [Turney, 2000], which aims at determining the most important terms in a document. The resulting keyphrases are assumed to reflect the document topic, and they are typically used for summarization, clustering, or search. Another application, which is gaining importance due to the popularity of Web 2.0, is product feature extraction [Yi *et al.*, 2003], which is often performed on customer reviews. Here, identified product features are used to create feature-oriented summaries of customer review collections, or as the basis for extracting opinions about features. Keyphrase extraction and product feature extraction mainly differ in their definition of "relevance". In keyphrase extraction, the goal is to identify those terms in a given document which best describe its topic by distinguishing it from documents with different topics. Individual *mentions* of the same term are not considered. In product feature extraction, on the other hand, the goal is to extract *all mentions* of features of a given product. At the same time, it is important to only extract features of the product *under review*, and not of any other products mentioned e.g. in comparisons. Approaches which rely on statistical information have been successfully employed for both applications in previous research [Tomokiyo and Hurst, 2003; Yi *et al.*, 2003]. Both of the above approaches utilize the Likelihood Ratio Test [Dunning, 1993] (LRT), which is well suited for the identification of relevant terms from document collections, since it does not assume a normal distribution of the variables (= frequencies of the terms) in the data. LRT compares the frequencies of candidate terms in the document collection to be analyzed with their frequencies in a general-language corpus and calculates the likelihood that a term is relevant for the given document collection.

For product feature extraction in customer reviews, other statistical methods have also been used: Hu and Liu [2004] try to summarize customer reviews and present a system which uses association mining to extract product features in opinionated sentences. They only present an evaluation of the combined product feature extraction and opinion mining steps, and report an average F-measure of 0.79 with the best configuration. One advantage of their system is that it does not rely on any pre-built knowledge base, but only uses statistical information. Popescu and Etzioni [2005] employ Point-wise Mutual Information to compute the probability that a candidate term is a feature of a given product, and they report an average F-measure of 0.758. Since the calculation of Point-wise Mutual Information requires a very large corpus, they use the web and a web search engine. Wong et al. [2008] model product feature extraction as a Dirichlet process prior which they then use in an Expectation Maximization algorithm. They reach an average F-measure between 0.58 and 0.95 on their four datasets. The *Sentiment Analyzer* system by Yi et al. [2003] includes a product feature extraction component utilizing base noun phrase patterns and LRT. For their product feature extraction step (on two datasets), Yi et al. [2003] report precision values of 0.97 and 1.0, but no recall. In [Ferreira *et al.*, 2008] we present a comparative evaluation of the approaches by Hu and Liu [2004] and Yi et al. [2003], this time evaluating product feature extraction independently of opinion detection. They find two limitations of LRT: 1) it often fails to identify **rare** product features, and 2) it also often fails to identify terms that are **both product features and general vocabulary items** (e.g. *weight*, *speed*, *option*).

This paper introduces LRT$_{wiki}$, our extension of the Likelihood Ratio Test algorithm, which addresses the above limitations by enriching LRT with encyclopedic information drawn from Wikipedia. In LRT$_{wiki}$, Wikipedia is employed as a

[1] We use *term* here to cover both single terms and multi-term expressions.

general-purpose source of domain knowledge. We analyze the performance of LRT$_{wiki}$ in two different tasks: In the first scenario we employ the algorithm for product feature extraction as in [Ferreira *et al.*, 2008], in the second scenario we employ it for keyphrase extraction as in [Tomokiyo and Hurst, 2003]. The remainder of this paper is structured as follows: Section 2 gives an overview of the data we use in our experiments. Section 3 describes LRT$_{wiki}$, our proposed extension of LRT, and Section 4 contains the experimental results and discussion. Conclusions and future work can be found in Section 5.

2 Employed Corpora

2.1 Data for Product Feature Extraction

We use the dataset as in [Ferreira *et al.*, 2008], for which a corpus originally annotated by [Hu and Liu, 2004] was reannotated. In contrast to the original annotations, in [Ferreira *et al.*, 2008] we annotated *all* mentions of product features, irrespective of there being an opinion expressed about them. Table 1 outlines some statistics on the dataset. We did not report any inter-annotator agreement statistics before, but since we are interested in the agreement (e.g. as an upper bound for the evaluation of the product feature extraction task), we reannotated a subset of the corpus in a controlled manner. First, we randomly selected 60 sentences from each of the five product review sets, and then we had two human subjects annotate them following the guidelines presented in [Ferreira *et al.*, 2008]. Due to the skewed class distribution (the vast majority of terms in product reviews are *not* product features), we simply calculated Precision, Recall, and F-measure (instead of e.g. Kappa) on the overlap between the two annotators. The overlap was calculated rather strictly by considering only exact matches in the product feature annotation. The results of the annotation overlap measurements are shown in Table 2.

Table 1: Product review datasets

Dataset	Documents	Sentences	Feature Mentions
Digital camera 1 (DC1)	45	597	594
Digital camera 2 (DC2)	34	346	340
Cell phone (CP)	41	546	471
MP3 player (MP3)	95	1716	1031
DVD player (DVD)	99	739	519

Table 2: Annotation overlap for product feature mentions

Dataset	Sentences	Words	Features	F-measure
DC1	60	980	67	0.736
DC2	60	1029	69	0.747
CP	60	1001	63	0.825
MP3	60	830	46	0.745
DVD	60	883	52	0.477

An analysis of the annotation overlap shows that product feature extraction is not a trivial task. F-measure on the DVD player dataset is particularly low. We observe that this is due to excessive usage of abbreviations regarding the product in this document collection (e.g. referring to the product with just its model number) and some disagreement regarding their annotation.

2.2 Data for Keyphrase Extraction

The data we employ in our keyphrase extraction experiments is originally from the DUC2001 dataset [Over, 2001]. The corpus consists of 309 news articles with keyphrases annotated by Wan and Xiao [2008]. The articles cover 30 different news topics and have an average length of 740 words. The annotation involved two annotators, who were allowed to select a maximum of 10 distinct keyphrases per document. Wan and Xiao report an inter-annotator agreement of 0.70 κ. After the annotation, the annotators created the final gold standard by resolving conflicting annotations in a discussion. The average number of keyphrases per document is 8.08, and the average number of words per keyphrase is 2.09 [Wan and Xiao, 2008]. Since LRT requires a collection of "on-topic" documents for extracting the most relevant terms, we selected the two largest subsets of the DUC2001 datasets. Each of the two subsets (DUC IDs: d06a & d34f) contains 16 documents.

3 LRT and LRT$_{wiki}$

3.1 LRT

LRT was introduced by Dunning [1993] and has been employed for many different NLP-related tasks, since the algorithm does not assume that the population it operates on is distributed normally or approximately normally, which is true for the frequencies of terms in a text. In short, LRT identifies relevant terms from a document collection by comparing the frequencies of the candidate terms in the "on-topic" documents with their frequencies in a general language "off-topic" document collection. It uses a contingency table for the current candidate term T, for which the frequency related values C_{11} to C_{22} are extracted from the "on-topic" document collection D_+ and the "off-topic" document collection D_-. Table 3 outlines the different elements of the contingency table, the LRT definition is shown in Equation (1).

Table 3: Contingency table for candidate term T

	D_+	D_-
T	C_{11}	C_{12}
$\overline{T}$	C_{21}	C_{22}

$$r_1 = \frac{C_{11}}{C_{11}+C_{12}}, r_2 = \frac{C_{21}}{C_{21}+C_{22}}, r = \frac{C_{11}+C_{21}}{C_{11}+C_{12}+C_{21}+C_{22}}$$

$$\begin{aligned} lr =& (C_{11}+C_{21})\log(r) + (C_{12}+C_{22})\log(1-r) - C_{11}\log(r_1) \\ & - C_{12}\log(1-r_1) - C_{21}\log(r_2) - C_{22}\log(1-r_2) \end{aligned}$$

$$-2\log\lambda = \begin{cases} -2 * lr & \text{if } r_2 < r_1 \\ 0 & \text{if } r_2 \geq r_1 \end{cases} \quad (1)$$

In their application of LRT to product feature extraction, Yi et al. [2003] and Ferreira et al. [2008] report high precision but low recall. As already described in Section 1, in [Ferreira *et al.*, 2008] we observe that LRT typically misses product features that have a low frequency in the "on-topic" document collection - e.g. because only very few customers comment on them - even if they do not occur in the "off-topic" document collection at all. In addition, LRT also misses terms which are both product features and general vocabulary items, such as *speed, option, flexibility*. Section 3.2 describes LRT$_{wiki}$, which is our proposed enhancement of LRT specifically targetting these two shortcomings.

3.2 LRT$_{wiki}$

LRT$_{wiki}$ aims at improving the ranking of candidate terms of two problematic candidate classes:

1. Candidate terms which occur in the "on-topic" document collection with low frequency and not at all or with a low frequency in the "off-topic" document collection
2. Candidate terms which occur both in the "on-" and "off-topic" document collection with medium or high frequency

The central idea of LRT$_{wiki}$ is to employ a comprehensive domain specific knowledge source containing the terminology typically used in the current domain as the source of additional "on-topic" data sets, and to modify the calculation of the LRT algorithm to take advantage of this new knowledge source. The knowledge source is created on the basis of Wikipedia.

Wikipedia Data

We chose the free online encyclopedia Wikipedia for two reasons: 1) Due to its broad coverage, we can expect it to contain articles about many topics. 2) Due to the encyclopedic style of Wikipedia articles, they tend to focus on a single topic, and normally do not contain irrelevant information.

Since our goal is to extract an additional corpus about the pre-defined topic(s) dealt with in the document collection to be analysed, we assume the topic to be known in advance. We then query Wikipedia in order to retrieve one article for each topic as the seed for retrieving the new "on-topic" data sets. For the product classes from the Hu and Liu dataset, we used the names provided in their paper (*digital camera*, *dvd player*, *mp3 player*, *cell phone*). For each of these topics there is either a Wikipedia article with the same title or an automatic redirect page (*mp3 player* → *Digital audio player*). For the DUC data, we read the documents and inferred the topics "police brutality" (d06a), for which Wikipedia contains an article, and "atlantic hurricanes" (d34f), which is redirected to "North Atlantic tropical cyclone". The Wikipedia-based document collection for each topic is built by extracting the categories to which the seed article belongs and then extracting all articles found in these categories. We performed a simple ad-hoc filtering only: We ignored all subcategories of "Wikipedia administration", since they do not carry any semantic content, and all categories with more than 200 articles, since we regard them as too broad. The article pages retrieved in this manner were then automatically cleaned of all Wikipedia markup, metainformation, references, and hyperlinks. The data we retrieved is extracted from a Wikipedia dump from Feb. 2007. Some statistics about the resulting data is given in Table 4.

Table 4: Content retrieved from Wikipedia

Wikipedia Seed Article	Retrieved Articles	Word Count
digital camera	263	161459
cell phone	250	204410
digital audio player	64	79099
dvd player	100	99898
north atlantic tropical cyclone	403	459046
police brutality	166	127216

Modifying the LRT Algorithm

The new Wikipedia content provides an additional document collection D_W on the basis of which we can calculate C_{13} for a given term T, which are defined in analogy to Table 3. With these new values we modify the calculation of the original LRT lr as follows:

$$\begin{aligned} lr_{mod} =& (C_{11mod} + C_{21}) \log(r) + (C_{12mod} + C_{22}) \log(1-r) \\ & - C_{11mod} \log(r_1) - C_{12mod} \log(1-r_1) - C_{21} \log(r_2) \\ & - C_{22} \log(1-r_2) \end{aligned}$$

$$C_{11mod} = \begin{cases} C_{11} + C_{13} & \text{if } C_{11} < t_1 \text{ and } C_{12} < t_1 \\ C_{11} + C_{13} & \text{if } C_{11} > t_2 \text{ and } C_{12} > t_2 \end{cases}$$

$$C_{12mod} = \begin{cases} 0 & \text{if } C_{11} < t_1 \text{ and } C_{12} < t_1 \\ \max(0, C_{12} - C_{13}) & \text{if } C_{11} > t_2 \text{ and } C_{12} > t_2 \end{cases} \quad (2)$$

The two thresholds t_1 and t_2 are used to set the boundaries for terms with low frequency (t_1) and terms with medium or high frequency (t_2).

4 Experiments

We model term extraction as a two-step process. In the first step, candidate terms are extracted, and in the second step, these candidates are ranked on the basis of their LRT / LRT$_{wiki}$ values. An overview of our architecture is shown in Figure 1. As in [Ferreira *et al.*, 2008], we employed the 600 randomly selected documents from the UKWaC British English web corpus [Ferraresi *et al.*, 2008] as an "off-topic" corpus (D_-).

4.1 Product Feature Extraction

For the product feature extraction evaluation, we follow the approach of Yi et al. [2003]. They specify their candidate patterns as the following Base Noun Phrases (BNP): NN, NN NN, JJ NN, NN NN NN, JJ NN NN, JJ JJ NN. These patterns are applied inclusively, i.e. multi-term expressions (e.g. *digital camera*) are allowed to be matched more than once. This way, occurrences of terms as parts of multi-term expressions (e.g. *camera*) are counted several times, thus boosting the extraction of the embedded term. After extracting the candidate terms, both LRT versions calculate the likelihood score for

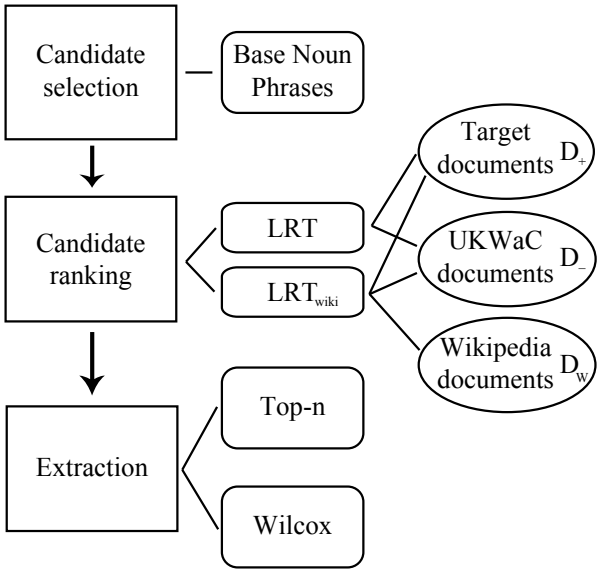

Figure 1: Term extraction architecture

each. The resulting ranked list has then to be transformed into a set of relevant and irrelevant features. Yi et al. [2003] address this issue by selecting the top n features as relevant, where n is the number of candidate features selected by the most restrictive BNP pattern set of "beginning definite Base Noun Phrases" [Yi *et al.*, 2003]. While this approach can lead to a high precision extraction, it unfortunately suffers from an extremely low recall. Therefore, we propose a different method for selecting the threshold for distinguishing relevant from irrelevant terms, which is based on the algorithm for outlier detection presented in [Wilcox, 2001, page 38]. According to this method, the threshold t_{LRT} for feature extraction is set to:

$$t_{LRT} = m_{lr} + sd_{lr} \quad (3)$$

where m_{lr} is the mean likelihood value and sd_{lr} is the standard deviation.

Table 5 shows the results obtained in [Ferreira *et al.*, 2008] and our experiments. The column "LRT Wilcox Threshold" shows the effect of the dynamic threshold calculation while using our reimplementation of the original LRT algorithm. The column "LRT$_{wiki}$ Wilcox Threshold" shows the results obtained by employing the dynamic threshold calculation and LRT$_{wiki}$. Following [Ferreira *et al.*, 2008], we perform an evaluation on each mention of a product feature, comparing the lowercased and lemmatized forms of the automatically extracted features with those in the gold standard. For LRT$_{wiki}$, we set t_1 to 5, which was empirically defined in order to reflect a threshold under which we consider a term to be rare. Likewise, the threshold t_2 was set to 10, meaning we consider terms which are found more often to be frequently occurring. These thresholds are optimal to the corpora we experimented with while smaller or larger values might make sense for different input corpora D_+.

4.2 Keyphrase Extraction

As we are interested in an state-of-the-art approach for keyphrase extraction which is also unsupervised, we employ the TextRank system [Mihalcea and Tarau, 2004]. We follow Mihalcea & Tarau by selecting only adjectives and nouns as candidate terms. The matching is done in a greedy fashion on the terms' POS tags with the following regular expression: $(JJ|JJR|JJS)^*(NN|NNS|NP|NPS)^+$. Greedy matching makes sure that only the longest matching phrases in a sentence are selected as candidates. This matching strategy is based on the observation that *complete* noun phrases are typically annotated as keyphrases in the DUC data set. For example "accidental shooting death" is annotated as a keyphrase and not just "shooting death" or "death". Contrary to product features, there is no clear-cut definition of what is and what is not regarded as a keyphrase for a document. Therefore, during our evaluation, we did not employ a threshold like in 4.1. Alternatively, we evaluate Precision, Recall and F-measure of the *top-n* extracted keyphrases. As a baseline system we employ TextRank in its default configuration. The keyphrases extracted by the TextRank system, the two versions of LRT, and the keyphrases in the gold standard are lemmatized and lowercased before comparison. When employing LRT$_{wiki}$, we use the same thresholds t_1 and t_2 as described in Section 4.1. We evaluate the top-n keyphrases (n $\leq$ 10) on the two datasets each containing 16 documents as described in Section 2.2. The results of the keyphrase extraction evaluation are shown in Table 6.

4.3 Error Analysis

As evident from Tables 5 and 6, LRT$_{wiki}$ consistently and significantly[2] improves F-measure in both applications. In the following, we perform an error analysis in the two applications separately.

Product Feature Extraction Error Analysis

When comparing the results of "LRT Wilcox" with "LRT in [Ferreira *et al.*, 2008]", one can already observe a constant improvement in precision and recall. This shows that the extraction strategy is also an important aspect of the LRT which might deserve further research. In the task of product feature extraction, the recall slightly decreases when comparing LRT and LRT$_{wiki}$ on two of the datasets (DC2, MP3). However, the concurrent gains in precision outweigh them, leading to an overall higher F-measure. The decrease of recall on some datasets can be explained as follows: A substantial amount of terms belonging to the specific vocabulary of the domain have C_{11} and C_{12} values smaller than t_1 and therefore receive a *boosting* from the new Wikipedia content. The boosting often pushes their lr_{mod} values into regions of other terms which have a $C_{11} > t_1$ and which would therefore typically be extracted as relevant. However, the boosting effect raises the overall average likelihood ratio, which we use to separate the relevant from the irrelevant terms. At the same time, there are typically quite a few terms which occur

[2]Significance of improvement in F-measure is tested using a paired one-tailed t-test and p $\leq$ 0.05 (*), p $\leq$ 0.01 (**), and p $\leq$ 0.005 (***)

Table 5: Product Feature Extraction

	LRT in [Ferreira *et al.*, 2008]			LRT Wilcox Threshold			LRT_{wiki} Wilcox Threshold			
Dataset	P	R	F	P	R	F	P	R	F	Δ F
DC1	0.671	0.495	0.570	0.750	0.513	0.609	0.760	0.574	0.654	**+0.045***
DC2	0.634	0.347	0.449	0.800	0.485	0.604	0.875	0.474	0.615	**+0.011***
CP	0.659	0.459	0.541	0.579	0.535	0.556	0.813	0.544	0.651	**+0.095***
MP3	0.339	0.408	0.370	0.513	0.665	0.579	0.560	0.661	0.606	**+0.027***
DVD	0.506	0.243	0.328	0.633	0.416	0.502	0.667	0.458	0.543	**+0.040***

Table 6: Keyphrase Extraction

		TextRank			LRT			LRT_{wiki}			
Dataset	**n**	P	R	F	P	R	F	P	R	F	Δ F
d06a	1	0.188	0.029	0.051	0.000	0.000	0.000	0.062	0.010	0.017	**+0.017*****
(16 docs)	2	0.125	0.039	0.060	0.094	0.030	0.045	0.375	0.119	0.180	**+0.135*****
	3	0.083	0.039	0.053	0.250	0.119	0.161	0.333	0.158	0.215	**+0.054*****
	4	0.094	0.059	0.072	0.281	0.178	0.218	0.328	0.208	0.255	**+0.037*****
	5	0.088	0.069	0.077	0.250	0.198	0.221	0.325	0.257	0.287	**+0.066*****
	6	0.073	0.069	0.071	0.250	0.238	0.244	0.312	0.297	0.305	**+0.061*****
	7	0.071	0.078	0.075	0.232	0.257	0.244	0.295	0.327	0.310	**+0.066*****
	8	0.070	0.088	0.078	0.234	0.297	0.262	0.273	0.347	0.306	**+0.044*****
	9	0.069	0.098	0.081	0.222	0.317	0.261	0.250	0.356	0.294	**+0.033*****
	10	0.075	0.118	0.092	0.219	0.347	0.268	0.238	0.376	0.291	**+0.023*****
d34f	1	0.467	0.053	0.096	0.062	0.008	0.014	0.062	0.008	0.014	**+0.000**
(16 docs)	2	0.500	0.115	0.186	0.062	0.015	0.024	0.062	0.015	0.024	**+0.000**
	3	0.500	0.168	0.251	0.042	0.015	0.022	0.062	0.023	0.033	**+0.011*****
	4	0.448	0.198	0.275	0.062	0.030	0.041	0.078	0.038	0.051	**+0.010*****
	5	0.431	0.237	0.305	0.100	0.061	0.075	0.150	0.091	0.113	**+0.038*****
	6	0.393	0.252	0.307	0.115	0.083	0.096	0.167	0.121	0.140	**+0.044*****
	7	0.396	0.290	0.335	0.125	0.106	0.115	0.170	0.144	0.156	**+0.041*****
	8	0.374	0.305	0.336	0.148	0.144	0.146	0.172	0.167	0.169	**+0.023*****
	9	0.350	0.313	0.331	0.139	0.152	0.145	0.167	0.182	0.174	**+0.029*****
	10	0.325	0.313	0.319	0.138	0.167	0.151	0.169	0.205	0.185	**+0.034*****

in almost every sentence (e.g. the product under review) and which therefore influence the standard deviation. In general, the boosting modification leads to a substantial increase in the average likelihood ratio, while hardly affecting the standard deviation. This leads to a slight increase in the threshold for the extraction of terms, with some relevant terms no longer reaching it. On the DC1 dataset, e.g. LRT_{wiki} extracts the correct product features "control", "film", and "sensor", while the original LRT misses them. At the same time, using LRT_{wiki}, the correct features "external flash" and "lcd screen" do not reach the threshold any more while the original LRT extracts them.

This effect on the average likelihood ratio (which is even more pronounced for the standard deviation) is caused by the modification which aims to improve the extraction of relevant terms also occurring in the general language corpus. Such candidates typically have a rather high C_{11} value, and due to the Wikipedia content their C_{12} value is reduced, leading to a very high likelihood ratio, which in turn leads to a higher standard deviation.

The inclusion of the Wikipedia documents also exacerbates one of the issues mentioned in [Ferreira *et al.*, 2008]: If several reviews mention products of another manufacturer or a different model (e.g. in comparisons), LRT (and also LRT_{wiki}) will extract them. Since the documents from Wikipedia are typically not limited to a single product, but rather to a product class, they tend to contain names of several different models and manufacturers. If such a name is mentioned in the "on-topic" documents, its likelihood ratio will be boosted due to our algorithm modification. Overall, however, LRT_{wiki} still leads to a significant improvement over LRT regarding F-measure.

Keyphrase Extraction Error Analysis

When comparing the results of the keyphrase extraction based on both versions of LRT with TextRank as a baseline, we observe that on the d06a dataset both LRT versions perform considerably better and on the d34f dataset considerably worse than the TextRank system. However, the performance of TextRank on the d34f dataset is much better than its average on the entire DUC2001 dataset: TextRank yields an overall F-measure of 0.132 at 10 extracted keyphrases on the entire DUC2001 dataset[3], while on the d34f dataset it reaches an F-Measure of 0.319 at 10 extracted keyphrases. This is due to the fact that, with three to five words, the keyphrases on the d34f subset are rather long compared to those in the other document sets (overall average keyphrase length in words:

[3]We obtained this result in a separate experiment.

2.0). When employed in a keyphrase extraction task, both versions of LRT have the limitation that the relevance of a term is calculated on the overall document collection. However, keyphrases are annotated with respect to their importance in individual documents. Therefore, LRT often fails to extract keyphrases which are only relevant for one document. This definition of relevance is different from the product feature extraction task, as terms are regarded as relevant over the entire document collection.

5 Conclusions and Future Work

In this paper we presented LRT_{wiki}, an enhancement of the Likelihood Ratio Test, which makes use of encyclopedic documents retrieved from Wikipedia. The enhanced algorithm leads to an improvement regarding the relevance ranking of terms. Since Wikipedia is available in many languages and its content is very broad, is seems to be a well suited resource for extending a statistical method for term mining. We also propose a method to calculate the threshold for the separation of relevant and irrelevant terms. In our empirical evaluation, LRT_{wiki} yielded a significant improvement regarding F-measure in two different tasks: keyphrase extraction and product feature extraction. A limitation which remains for product feature extraction regards terms which belong to the current domain, but which are not features of the product under review. Although somewhat less consistent, our results show that LRT_{wiki} might also be a promising candidate for keyphrase extraction. While our enhancements consistently improve the results regarding F-measure over the original LRT, both LRT and LRT_{wiki} are inferior to the non-LRT baseline on one of two datasets. In future work, we plan to further investigate this issue.

Other lines of future work include the following: We plan to improve the threshold calculation for the separation of relevant and irrelevant terms, since the current approach is optimized for the original LRT, and not yet for LRT_{wiki}. We will also address remaining issues regarding the product feature extraction task. For this, we plan to extract additional information regarding individual products (e.g. product and manufacturer names) from Wikipedia, and to use this information to filter out feature candidates not pertaining to the product under review. We plan to extract this additional information from *semi-structured* Wikipedia content (e.g. links, lists, and tables), thus going beyond treating Wikipedia articles as flat documents.

Acknowledgements

The project was funded by means of the German Federal Ministry of Economy and Technology under the promotional reference "01MQ07012". The authors take the responsibility for the contents.

References

[Dunning, 1993] Ted Dunning. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics*, 19(1):61–74, 1993.

[Ferraresi *et al.*, 2008] Adriano Ferraresi, Eros Zanchetta, Silvia Bernardini, and Marco Baroni. Introducing and evaluating ukwac, a very large web-derived corpus of english. In *Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can we beat Google?*, pages 47–54, Marrakech, Morocco, June 2008.

[Ferreira *et al.*, 2008] Liliana Ferreira, Niklas Jakob, and Iryna Gurevych. A comparative study of feature extraction algorithms in customer reviews. In *Proceedings of the 2nd IEEE International Conference on Semantic Computing*, pages 144–151, Santa Clara, California, USA, August 2008.

[Hu and Liu, 2004] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177, Seattle, Washington, USA, August 2004.

[Mihalcea and Tarau, 2004] Rada Mihalcea and Paul Tarau. TextRank: Bringing Order into Texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing 2004*, pages 404–411, Barcelona, Spain, July 2004.

[Over, 2001] Paul Over. Introduction to duc-2001: an intrinsic evaluation of generic news text summarization systems. In *DUC 2001 Workshop on Text Summarization*, New Orleans, Louisiana, USA, September 2001.

[Popescu and Etzioni, 2005] Ana-Maria Popescu and Oren Etzioni. Extracting product features and opinions from reviews. In *Proceedings of the Human Language Technology Conference/Conference on Empirical Methods in Natural Language Processing*, pages 339–346, Vancouver, Canada, October 2005.

[Tomokiyo and Hurst, 2003] Takashi Tomokiyo and Matthew Hurst. A language model approach to keyphrase extraction. In *Proceedings of the ACL 2003 Workshop on Multiword Expressions: Analysis, Acquisition and Treatment*, pages 33–40, Sapporo, Japan, July 2003.

[Turney, 2000] Peter D. Turney. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2:303–336, 2000.

[Wan and Xiao, 2008] Xiaojun Wan and Jianguo Xiao. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pages 855–860, Chicago, Illinois, USA, July 2008.

[Wilcox, 2001] Rand R. Wilcox. *Fundamentals of Modern Statistical Methods: Substantially Improving Power and Accuracy*. Springer, 2001.

[Wong *et al.*, 2008] Tak-Lam Wong, Wai Lam, and Tik-Shun Wong. 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*, pages 35–42, Singapore, July 2008.

[Yi *et al.*, 2003] Jeonghee Yi, Tetsuya Nasukawa, Razvan C. Bunescu, and Wayne Niblack. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In *Proceedings of the 3rd IEEE International Conference on Data Mining*, pages 427–434, Melbourne, Florida, USA, December 2003.